UNIVERSITE PARIS DIDEROT (PARIS 7)

Antelope, une plate-forme de TAL permettant d'extraire les sens du texte

Théorie et applications de l'interface syntaxe-sémantique

THESE

pour obtenir le grade de

Docteur de l'Université Paris Diderot (Paris 7)

Discipline : linguistique théorique, descriptive et automatique

présentée et soutenue publiquement le 25 septembre 2012 par

François-Régis CHAUMARTIN

Membres du jury :

Présidente	**Laurence DANLOS**, Professeur des universités	Université Paris Diderot (Paris 7)
Rapporteurs	**Adeline NAZARENKO**, Professeur des universités **Pierre ZWEIGENBAUM**, Directeur de recherche	LIPN–Université Paris Nord (Paris 13) LIMSI–CNRS
Examinateurs	**Christian JACQUELINET**, Médecin des hôpitaux **Guy PERRIER**, Professeur des universités	Agence de la Biomédecine, Lim&Bio LORIA–University Nancy 2
Directeur de thèse	**Sylvain KAHANE**, Professeur des universités	Université Paris Ouest Nanterre



MOTS-CLÉS : plate-forme linguistique, traitement de corpus, analyse syntaxique et sémantique, lexique sémantique, désambiguïsation lexicale, résolution d'anaphores et de coréférences, Théorie Sens-Texte, interface syntaxe-sémantique, extraction d'information, paraphrases, ontologies, Web sémantique.

KEYWORDS: linguistic platform, corpora analysis, parsing, semantic role labeling, semantic lexicon, word sense disambiguation, anaphora and coreference resolution, Meaning-Text Theory, syntax-semantics interface, Information Extraction, paraphrases, ontologies, Semantic Web.

Préface

Réaliser une thèse est une aventure, développer une entreprise en est une autre. J'ai la chance de pouvoir concilier deux passions exigeantes, l'innovation et l'entreprenariat. Mes parents étant de purs littéraires, mon choix d'études s'est logiquement porté à l'adolescence sur les mathématiques et l'informatique. Un diplôme d'ingénieur en poche, j'ai créé en 1994 une première société dédiée au service et au conseil informatique. Notre équipe a mené pendant douze ans des projets innovants pour le compte de grands clients. Je citerai ici trois projets significatifs, qui ont exercé une influence sur mon parcours et m'ont indirectement conduit au choix incongru de démarrer une thèse à 37 ans :

- De 1997 à 2002, nous avons réalisé avec succès la partie technologique d'une refonte complète d'un système d'information (projet global à 50 000 jours-hommes). Nous avons conçu et implémenté une architecture logicielle ambitieuse, en avance sur son temps[1] et compatible avec le standard d'alors du développement logiciel en entreprise (COM+ de Microsoft). Ce système a donné par la suite un important avantage concurrentiel à notre client, en lui permettant de développer rapidement de nouveaux produits et en facilitant l'intégration d'autres systèmes d'information suite à des fusions et acquisitions.
- Fin 2000, Microsoft annonçait une nouvelle architecture de développement d'entreprise (.NET) faisant table rase du passé, un nouveau langage (C#) et une bibliothèque très complète de classes prêtes à l'emploi. Nous avons jugé l'ensemble innovant, élégant et efficace, et l'avons rapidement adopté. A ma connaissance, nous avons été les premiers en France à mettre en production un site de commerce électronique développé en .NET, en septembre 2001.
- En 2003, nous avons réalisé pour le compte de l'agence de biomédecine une application Web « méta-thésaurus de la greffe ». Son objectif était de qualifier plus finement les patients, organes et maladies, afin de fluidifier la recherche de l'organe le mieux adapté à un receveur particulier, et donc de sauver plus de vies chaque année.

En plus de l'intérêt sociétal, ce dernier projet a eu un impact particulier car c'était ma première rencontre avec le TAL et les ontologies. Je découvrais un univers informatique encore plus complexe que celui des systèmes d'information classiques. Au fil de nos conversations, mon interlocuteur de l'agence de biomédecine, titulaire d'un double doctorat en médecine et en linguistique informatique, m'a glissé : « *mais pourquoi ne pas faire une thèse en TAL ?* ». Ainsi fut semée la graine qui m'aura conduit à renoncer volontairement à bien des loisirs au profit de la recherche pendant sept ans.

En 2004, ma première société comptait une vingtaine de personnes. Après une période de lourdes turbulences (explosion de la bulle Internet, crise économique suite aux attentats du 11 septembre et aux conflits subséquents, dépôt de bilan de quelques clients), ma vie professionnelle était redevenue plus facile. Mais je m'ennuyais.

[1] Modélisation orientée objet du métier, génération automatique de code, persistance des objets en base relationnelle, bus logiciel asynchrone…

J'avais commencé à découvrir en autodidacte des projets comme WordNet et le Link Grammar. Le manque de liant entre ces différentes briques logicielles m'intriguait. Je déplorais de ne pas arriver à les intégrer et les manipuler facilement avec une boîte à outils comme celles permettant de construire des interfaces ou des bases de données. Ce manque m'a poussé à créer progressivement les bases de ce qui allait devenir la plate-forme Antelope.

Venant du monde de l'industrie, j'avoue rétrospectivement une certaine ignorance à cette époque de ce qu'est la recherche académique. Un matin de 2004, je prends mon bâton de pèlerin et vais toquer à la porte[2] de la patronne d'un laboratoire de TAL. La discussion est courte. Moi, tout sourire : « *Bonjour, je m'appelle François, je suis ingénieur et je voudrais faire une thèse* » ; elle, index pointant la porte : « *Dehors !* ». Après ce premier échange encourageant et l'envoi d'une longue lettre de motivation, je démarre un master de recherche en TAL en parallèle à mon activité principale, en jonglant avec l'agenda et en apprenant à intercaler un TD de λ-calcul entre deux rendez-vous professionnels. Après le master, j'embraye en thèse et passe la main sur ma première société fin 2006, notamment pour dégager plus de temps pour la recherche. Mais comme la vie entrepreneuriale me manque, je repars de zéro début 2007 en créant Proxem (pour ***proc****édures* ***sém****antiques*). Depuis, je mène de front les deux aventures, en consacrant un temps partiel à la R&D.

Du fait de la cohabitation de ces deux activités parallèles, la réalisation de cette thèse s'est étendue sur une période plus longue que souhaitée, ponctuée aussi par la naissance de ma troisième fille. Cela n'est pas sans poser souci, car la fraîcheur des travaux de recherche ne dure qu'un temps. Les miens se placent résolument dans une perspective d'applications concrètes du TAL, en y appliquant l'expérience accumulée en plus de 15 ans de développement informatique innovant. J'ai souvent eu le sentiment de faire le grand écart entre la démarche d'un ingénieur (qui commence par collecter avec une démarche holistique tout ce qui marche, en le complétant pour que ça marche mieux) et celle d'un chercheur (qui a tendance à rechercher les problèmes les plus fins possibles à résoudre), au risque de parfois confondre ce qui relève du travail de recherche avec l'activité d'ingénierie. Au final, la partie visible de mon activité de recherche consiste en huit publications (conférences TALN, RECITAL, GSCL, ACL workshop Sem-Eval, RIAO, RMLL) et deux articles (revues TAL et JLCL), avec le plaisir de quelques collaborations académiques.

Chez Proxem, j'ai la chance d'être entouré d'une équipe talentueuse, dont les membres ont repris mes travaux pour les industrialiser. Pour la suite, je laisserai la parole au *nous* d'auteur ; quand il s'agira de travaux que je n'ai pas directement réalisés, j'utiliserai l'expression *l'équipe Proxem*.

-o-

Cette thèse est dédiée à la mémoire de mon père, François-Régis Chaumartin Sr. (1934 – 2012), agrégé de grammaire, docteur ès lettres classiques, professeur de latin à l'université de Dakar puis à l'université Paris-Est Créteil. Mon père est parti hélas trop tôt, le 25 août 2012, un mois exactement avant ma soutenance de thèse, à laquelle il aurait tant souhaité assister, et dont il aura été un relecteur assidu. Il a gardé jusqu'au dernier jour son intelligence, sa vivacité d'esprit et sa mémoire extraordinaire, qui était aussi en partie la mienne.

---

[2] Pour la petite histoire, les locaux du laboratoire étaient temporairement dans l'immeuble du somptueux siège social de RFF. Sur l'instant, le manque de moyens parfois déploré par les chercheurs m'avait semblé très relatif.

Remerciements

J'adresse mes remerciements à celles et ceux qui ont facilité la naissance de ce travail de recherche et favorisé son bon aboutissement. Ils vont, dans un ordre essentiellement chronologique, à :

Mes parents, qui m'ont immergé dans un environnement littéraire et transmis le goût des mots, mais qui m'ont poussé vers les études scientifiques qui fournissent un cadre formel pour les comprendre,

Les créateurs d'ELIZA, FRUMP, HAL et d'autres programmes spectaculaires qui donnent du rêve,

Bruno Petazzoni, Professeur de mathématiques et d'informatique, pour ses enseignements dispensés pendant mes jeunes années,

Christian Jacquelinet, Docteur en médecine et en linguistique informatique, qui m'a donné l'idée de reprendre la voie des études via la recherche, douze ans après mon diplôme d'ingénieur,

Laurence Danlos, Professeur de linguistique informatique à l'Université Paris Diderot – Paris 7, qui a accueilli et soutenu un étudiant atypique dans son master de recherche puis dans son laboratoire,

Sylvain Kahane, Professeur à l'Université Paris Ouest Nanterre, qui jongle avec les constituants de la langue avec la dextérité du linguiste et la rigueur du mathématicien, dont j'ai éprouvé la patience,

Pierre Zweigenbaum et Adeline Nazarenko, qui ont accepté d'être rapporteurs de cette thèse,

Guy Perrier, membre du jury, avec qui les conversations sur l'ISS sont toujours passionnantes,

Les 2 500 internautes ayant téléchargé et utilisé Antelope, qui ont contribué au projet par leurs avis,

Les stagiaires que j'ai encadrés chez Proxem et qui ont participé directement ou indirectement à la plate-forme : Benjamin Surma, Ricardo Minhoto, Julien Jacquelinet, Stéphanie Paina, Jean-Damien Hatzenbuhler, Remi Takase, Joanne Boisson et Roxane Anquetil.

L'équipe Proxem, qui apprécie d'autant plus la R&D quand elle s'applique aux projets concrets : Jocelyn Coulmance, Nicolas Frelat, Etienne Coumont, Amélie Cochet-Grasset, Fanny Parganin, Paul Bédaride ; Eglantine Schmitt et Eric Vernet pour leur point de vue *digital humanities* et commercial.

Celles et ceux qui ont eu la gentillesse de relire ce document et de formuler commentaires, remarques et critiques constructives,

Last but not least, Carole, mon indéfectible soutien, et Cerise, Émilie et Mahaut, à qui j'ai volé trop de temps ces dernières années.

Contenu

Liste des figures

Liste des tableaux

Partie I. Introduction

A. Pourquoi une plate-forme de TAL ?

Une application informatique vise à rendre un service à des utilisateurs, humains ou autres systèmes informatiques, en automatisant un processus de traitement de données. Les applications classiques manipulent des données structurées (parfois avec des volumétries très importantes) avec des algorithmes déterministes. Elles représentent la grande majorité des systèmes actuels : programmes de gestion, jeux, suites bureautiques… Les notions de précision, de rappel ou de F-mesure ont peu de sens dans ce contexte. En effet, un même jeu de données fourni en entrée produira en principe toujours le même résultat ; et si ces données sont correctes, les résultats le seront aussi, si l'implémentation est exempte de bugs. L'indicateur de qualité du traitement d'une tâche sera plutôt son temps d'exécution, par exemple.

La résolution de certains problèmes nécessite de faire preuve d'*intelligence* ; cette notion est sujette à de multiples interprétations, et nous ne chercherons pas à la définir formellement. Un programme rentrant dans la catégorie « intelligence artificielle » cherche à résoudre des problèmes auxquels même un humain ne trouve pas forcément une solution. Il s'agit typiquement de situations où il faut effectuer un choix sous un certain nombre de contraintes, parfois sans être assuré de l'existence d'une solution optimale ; ou encore de conditions complexes pour lesquelles les analystes humains peinent à expliciter un algorithme satisfaisant. La reconnaissance de formes dans des images et la résolution de problèmes d'échecs ou de recherche opérationnelle rentrent par exemple dans cette catégorie, ainsi que le Traitement Automatique des Langues.

Les frontières ne sont, à l'évidence, pas étanches entre ces deux catégories d'application. Des applications de logistique intègrent des modules de recherche opérationnelle pour résoudre des problèmes d'optimisation ; les traitements de texte proposent des correcteurs d'orthographe et de grammaire ; les bases de données utilisent des heuristiques d'optimisation complexes pour traduire les requêtes en opérations élémentaires.

Est-il plus facile de développer une application informatique classique ou un programme d'intelligence artificielle ? La réponse n'est pas si simple. En effet, les applications classiques ont une complexité grandissante et il ne faut pas minimiser l'effort nécessaire pour les développer. Elles comptent parfois des millions de lignes de code, reflet des exigences croissantes des demandes d'utilisateurs ; par exemple, un système de calcul de retraite doit implémenter des règles de gestion complexes pour gérer des historiques de carrières sur plus de 40 ans, sous la contrainte d'une législation qui évolue avec le temps.

Néanmoins, cette complexité est aujourd'hui maîtrisable aux différents maillons de la chaîne de développement : analyse du problème, conception du système, implémentation, tests, production, maintenance. Au fil des années, des méthodes sont apparues pour théoriser et rationaliser ces différentes phases ; des guides méthodologiques offrent un cadre de travail et proposent des

réponses standards aux problèmes les plus fréquents. L'arrivée de plates-formes de développement[3] et de boîtes à outils (*frameworks*) de composants prêts à l'emploi a permis d'accroître la productivité et le confort des développeurs d'applications, et d'asseoir progressivement une industrie du logiciel.

Il nous semble que le TAL n'a pour l'instant bénéficié que modestement des contributions du génie logiciel et de l'industrialisation des développements informatiques. Cette situation nous semble être due à la conjonction de plusieurs facteurs :

- La spécificité du TAL est de **cumuler un grand nombre de tâches complexes et de problèmes non résolus à ce jour** : résolution d'anaphores, désambiguïsation lexicale, correction orthographique, prise en compte des figures de styles… Cette complexité résulte des nombreuses ambiguïtés présentes dans les langues naturelles. Les acteurs du TAL se focalisent donc en premier lieu sur la résolution de problèmes unitaires, relevant souvent de la recherche fondamentale. Ceux-ci viennent en outre de disciplines variées (informatique théorique, mathématique, linguistique, psychologie cognitive, enseignement des langues…) parfois très éloignées du génie logiciel. L'industrialisation[4] des applications de TAL ne se fait donc que progressivement.
- Les normes[5] et standards[6] représentent un facteur important d'harmonisation pour une industrie donnée. Or, s'ils sont abondants en informatique, leur nombre reste faible dans le domaine du TAL. On peut néanmoins citer UIMA[7], les normes approuvées par l'ISO[8], ou encore les jeux d'étiquettes des *treebanks* largement diffusés.
- Les plates-formes[9] sont tout aussi essentielles pour structurer une industrie ; elles revendiquent d'ailleurs souvent une compatibilité avec telle norme ou tel standard. Or, il en existe relativement peu dédiées au TAL (on peut toutefois citer GATE, LingPipe ou OpenNLP comme architectures logicielles permettant de fédérer des composants de traitement).

L'industriel, le chercheur ou l'étudiant qui souhaite implémenter un algorithme de TAL, ou développer une application complète, consacre aujourd'hui une partie significative de son temps à résoudre des problèmes techniques sans grande valeur ajoutée. Quel langage de programmation utiliser ? Avec quel jeu de composants ? Comment les faire communiquer entre eux ? Comment passer facilement d'une langue à une autre ?

---

[3] Eclipse dans l'univers Java, Visual Studio dans le monde Microsoft, pour ne citer que les plus connues.
[4] Amélioration de la robustesse, élargissement de la couverture, capacité de passage à l'échelle…
[5] L'ISO définit une norme comme un « *document établi par consensus et approuvé par un organisme reconnu, qui fournit, pour des usages communs et répétés, des règles, des lignes directrices ou des caractéristiques, pour des activités ou leurs résultats garantissant un niveau d'ordre optimal dans un contexte donné* ».
[6] Un standard est un référentiel de large diffusion, consensuel, publié (par opposition à une norme) par une entité autre qu'un organisme de normalisation national ou international.
[7] Cf. la présentation d'UIMA page 35.
[8] Notamment au sein du groupe ISO/TC 37/SC4 : TMF (*Terminological Mark-up Framework*), norme ISO 16642, propose un méta-modèle comme cadre de représentation des bases de données terminologiques en XML ; la norme SynAF (*Syntactic annotation framework*) décrit un cadre d'annotation syntaxique.
[9] Une plate-forme logicielle propose une base technologique sur laquelle d'autres programmes peuvent être rapidement développés. C'est un système au sein duquel on peut utiliser et développer un ensemble de logiciels, et où des programmes applicatifs peuvent s'exécuter. Une plate-forme concerne généralement un contexte particulier : système d'exploitation, analyse d'images, calcul intensif, jeux vidéo… ou TAL. Les plates-formes sont généralement conçues, développées et maintenues par des acteurs informatiques de référence, car elles nécessitent un investissement important.

L'appel à communication de la revue TAL (2008, 49.2) consacrée aux *Plates-formes pour le traitement automatique des langues* résumait parfaitement ces problématiques : « *La recherche en Traitement Automatique des Langues fait de plus en plus souvent appel à des infrastructures logicielles complexes. Faute de modélisation « intégrative » du langage, on en produit des modélisations régionales, partielles, et une plate-forme est le moyen de les articuler entre elles, de les faire coopérer ; de ce fait, il est souvent nécessaire d'assembler au sein d'un même processus des traitements et des ressources de natures et de provenances diverses, ce qui pose d'importants problèmes d'interopérabilité. D'un autre côté, la complexité croissante des modèles linguistiques demande des moyens de formalisation sophistiqués tandis que la généralisation d'une approche expérimentale sur des corpus larges et de formats variés impose également des contraintes fortes sur les outils mis en œuvre.* »

Nous avons créé Antelope[10], une plate-forme industrielle de traitement du langage, pour apporter des réponses concrètes à ces problématiques : faciliter la résolution des problèmes purement informatiques, aider à maîtriser une complexité croissante et améliorer la productivité du développement en TAL. Nous allons à présent aborder plus précisément les problèmes que nous voulons résoudre, notamment les enjeux de la « compréhension » des textes.

B. Vers une meilleure compréhension des textes

La compréhension de textes est un domaine qui a périodiquement soulevé de grands espoirs, avant qu'ils ne retombent, les réalisations n'étant pas à la hauteur des attentes[11]. Toutefois, nous pensons nous rapprocher d'une situation où extraire le sens du texte deviendra un objectif réaliste. En effet, chaque grande vague a entraîné son lot de progrès. Des algorithmes nouveaux ont permis de progresser sur la plupart des tâches. Les techniques d'apprentissage automatique (*machine learning* en anglais) ont montré leur efficacité, en remplacement ou en complément des systèmes à connaissances expertes ; leur essor a été rendu possible par l'apparition d'un nombre croissant de corpus annotés manuellement, permettant cet apprentissage. La puissance de traitement des machines et leurs capacités de stockage ont régulièrement doublé. Nous sommes donc en présence d'une conjonction de facteurs favorables aux progrès dans ce domaine.

Notre objectif est d'**être capable de développer rapidement des applications de TAL** sachant « comprendre » des textes de différentes natures, écrits en anglais ou en français : articles de presse (dans une perspective de veille économique), textes encyclopédiques (pour extraire des connaissances sur le monde), verbatims de consommateurs (de façon à calculer un indice de satisfaction) ou encore documents RH (pour trouver les profils correspondant au mieux à une offre, par exemple). Nos travaux s'inscrivent donc résolument dans un cadre applicatif et opérationnel.

Notre ambition *in fine* est de **rendre calculable du texte tout-venant**. Plus précisément, nous souhaitons en calculer une représentation sémantique dont les éléments soient (au moins partiellement) désambiguïsés. Une telle représentation a de multiples intérêts et facilite la réalisation de tâches de haut niveau comme la traduction automatique ou le résumé de texte. Elle améliore aussi la qualité des informations qu'un utilisateur peut trouver sur Internet ; une compréhension fine de sa requête, dépassant le simple mot clé, permet alors d'améliorer la pertinence des résultats.

[10] Quasi acronyme pour *Advanced NaTural Language Object-oriented Processing Environment*.

[11] Le rapport Bar-Hillel concluait en 1960 à l'impossibilité d'une traduction automatique de qualité humaine.

(Nazarenko, 2004) propose la formulation suivante : « *De manière abstraite, on peut considérer que « comprendre un texte » signifie être capable de modifier sa représentation du monde en fonction des informations véhiculées par le texte. Cela suppose qu'un être humain ou un système intelligent dispose d'un ensemble de connaissances qui constitue sa vision de son environnement physique, intellectuel, social et symbolique. Dans cette perspective, la compréhension se traduit par l'ajout, la suppression ou la correction de connaissances. En pratique, le niveau de compréhension dépend de l'objectif visé et de la nature du texte considéré. On ne lit pas un texte de loi ou une police d'assurance comme un article de presse, un manuel scolaire comme une notice pharmaceutique. En soi, la compréhension n'est pas une tâche. C'est une activité préalable à de nombreuses tâches, comme le résumé, la traduction, l'exécution d'instructions…* »

C. Problématiques

La compréhension de textes soulève un grand nombre de difficultés, d'ordre théorique (que veut dire « comprendre » un texte ?), conceptuel (comment modéliser un énoncé complexe ?) et pratique (comment implémenter des algorithmes efficaces de TAL ?). Nous nous intéresserons ici aux deuxième et (surtout) troisième aspects. L'une des principales difficultés est de faire travailler conjointement plusieurs ressources, en autorisant leur assemblage rapide sous forme de composants ; nous précisons les difficultés liées à cette interopérabilité ci-dessous. D'autre part, nous nous inscrivons dans une perspective de développement rapide d'applications industrielles ; nous visons donc à réaliser des applications capables de monter en charge, robustes et performantes. Nous avons réalisé pour cela la plate-forme Antelope.

1. Rendre les ressources de TAL interopérables

Une large typologie de tâches d'analyse peut être effectuée sur des textes. Elles nécessitent deux types de ressources : des composants implémentant des algorithmes de traitement et des données linguistiques. La frontière entre les deux est parfois floue ; par exemple, le code d'un automate peut se ramener à un paramétrage.

Prise individuellement, chaque tâche d'analyse est intrinsèquement complexe, compte tenu de la nature par essence ambigüe de la langue. Pour la plupart des tâches, aucun algorithme prenant du texte « tout-venant » ne fonctionne à 100 %. Cette complexité est augmentée par le fait que ces tâches nécessitent des connaissances de plusieurs niveaux, que pourrait apporter une « compréhension » préalable du contexte, alors même que chaque tâche contribue à cette compréhension, au moins d'une façon parcellaire.

Mener plusieurs tâches d'analyse simultanément revient à faire coopérer différents composants dans une chaîne de traitement, et s'avère encore plus compliqué. En effet, un problème pratique en TAL provient du fait que les ressources sont généralement conçues et implémentées pour une tâche donnée, avec un formalisme dédié.

L'interopérabilité entre deux composants est la possibilité de leur faire analyser successivement un texte donné, en permettant au second composant d'utiliser les résultats du premier. Un exemple classique d'interopérabilité opérationnelle concerne un analyseur syntaxique travaillant sur la sortie d'un étiqueteur morphosyntaxique. Le prérequis ici est que les deux composants partagent le même jeu d'étiquettes (celui du *Penn Treebank* par exemple), faute de quoi leur dialogue est impossible.

Le même problème existe au niveau des données linguistiques : par exemple, un lexique aura deux sens du nom « chat » avec une description courte, tandis qu'un autre décrira trois sens avec une description longue, et des relations vers d'autres concepts. Quelle serait alors notre référence dans une tâche de désambiguïsation lexicale ?

Faute de disposer de protocoles partagés standards ou normés, et d'un modèle normalisé de représentation des informations linguistiques, la capacité d'interopérabilité entre ressources est donc loin d'être acquise. L'un de nos objectifs est de les rendre génériques et interchangeables pour une tâche donnée.

2. Simplifier le développement des applications de TAL

Chaque composant de traitement effectue une **tâche** précise, c'est-à-dire un fractionnement élémentaire du travail à fournir en vue de produire un résultat. Une **application** du TAL regroupe un ensemble de composants et de ressources pour aider un utilisateur (non nécessairement expert en traitement du langage) à faire un certain travail. La frontière entre ces deux notions est parfois floue ; certaines tâches de haut niveau ont une valeur perçue par l'utilisateur comme suffisante pour les promouvoir au rang d'applications à part entière : on peut citer par exemple la correction orthographique ou grammaticale[12].

Le développement des applications de TAL passe par l'intégration de plusieurs ressources. Cette interopérabilité est loin d'être immédiate, ce qui constitue un frein à leur implémentation.

D. Contributions

Notre objectif de recherche nécessitant la prise en compte simultanée de plusieurs tâches et ressources linguistiques ainsi que leur intégration, nous avons dû adopter une démarche « en largeur ». Une partie de nos travaux a porté aussi sur des tâches précises, avec une approche « en profondeur ». Au final, notre contribution directe porte essentiellement sur quatre points :

- La conception d'une **plate-forme industrielle de traitement du langage**, robuste et relativement simple à mettre en œuvre, qui permet différents niveaux de représentation. La plate-forme propose aussi un modèle de données linguistiques unifié. A notre connaissance, il n'existe pas d'autre plate-forme librement utilisable pour l'enseignement et la recherche aussi complète sur le jeu de composants fournis en standard.
- La constitution d'un **lexique sémantique multilingue à large couverture**, par intégration de données linguistiques d'origines diverses (WordNet, Wikipédia, SUMO…).
- La conception et l'implémentation de **composants d'analyse sémantique** dédiés à des tâches unitaires (reconnaissance d'entités nommées, extraction de relations, analyse d'opinion et de sentiments, résolution d'anaphores, regroupement de documents).
- Le prototypage d'une **interface syntaxe-sémantique** (ISS dans la suite) opérationnelle, rendu possible par la mise en commun des éléments précédents. Nous pensons avoir proposé une approche originale de cette interface, que nous qualifions d'approche *paresseuse*, dans la mesure où le calcul de ses règles est largement déduit d'exemples fournis par l'utilisateur.

---

[12] On retrouve cette dualité dans les traitements de texte : Word et Open Office proposent des fonctionnalités de ce type ; les applications commercialisées par Antidot et Synapse sont complètement dédiées à ces tâches.

Au final, Antelope facilite grandement le développement rapide d'applications de TAL. Ce point est illustré concrètement en partie VI, page 131, à travers la présentation de plusieurs applications qui ont été écrites avec la plate-forme.

E. Plan du document

Le plan du document, ci-dessous, présente plus en détail ces différentes contributions.

1. Notre cadre théorique

La partie II (page 9) propose différentes approches et formalismes de représentation du sens. Elle introduit celle que nous souhaitons obtenir, en établissant les liens nécessaires vers les données linguistiques à large couverture disponibles. Cette partie présente notamment les principes sous-jacents à notre plate-forme, inspirés par la Théorie Sens-Texte, qui postule des niveaux de représentation morphologique, syntaxique et sémantique.

2. La plate-forme Antelope

La partie III (page 19) présente la plate-forme Antelope, qui intègre et fédère différentes ressources linguistiques et des composants d'analyse syntaxique et sémantique.

Nous présentons brièvement les catégories de composants de traitement linguistique permettant d'effectuer les transitions entre niveaux de représentation, ainsi que la conception des échanges au sein de la plate-forme. La plate-forme propose un modèle de données linguistiques unifié, qui permet aux différentes tâches de partager leurs résultats. Ce modèle unifie les différents niveaux de représentation linguistique et autorise à préserver autant que possible les ambiguïtés au niveau lexical, syntaxique et sémantique.

L'architecture technique est aussi abordée ; nous présentons la conception informatique de la plate-forme, en insistant sur des « bonnes pratiques » de génie logiciel destinées à faciliter la modularité du logiciel et la réutilisation de composants. Cette partie expose aussi quelques caractéristiques d'Antelope : capacité de passage à l'échelle, présence d'un mécanisme d'extensibilité au niveau des principaux objets, intégration de composants externes écrits en divers langages.

Enfin, cette partie présente des plates-formes de TAL de référence, en positionnant Antelope par rapport à celles-ci. Soulignons que notre plate-forme est compatible avec l'architecture UIMA, qui est une norme et un standard ; elle peut donc être utilisée au sein de chaînes d'annotation faisant intervenir de multiples outils.

3. Intégration de données linguistiques à large couverture

La partie IV (page 41) montre comment l'intégration de plusieurs ressources à large couverture permet de créer un lexique sémantique multilingue ; ce lexique est centré sur une ressource bien connue dans le monde du TAL, le Princeton WordNet, qui sera présentée en détail. Cette intégration permet de pallier certaines insuffisances des ressources prises individuellement et montre comment elles se complètent.

Des expériences réalisées avec Antelope ont permis de produire des données linguistiques nouvelles qui enrichissent le lexique. Nous avons par exemple apparié des concepts de WordNet avec des articles de l'encyclopédie Wikipédia ; produit un catalogue de relations de polysémie régulière en n'utilisant que WordNet ; ou encore fait l'apprentissage de paraphrases à partir de paires d'articles encyclopédiques comparables.

4. Composants d'analyse sémantique

La partie V (page 89) détaille les composants d'analyse sémantique que nous avons spécifiquement développés dans le cadre de la plate-forme. Ils fournissent des résultats à l'état de l'art (c'était du moins le cas au moment de leur création) et sont mis en œuvre par l'ISS.

Ces composants traitent notamment des tâches de reconnaissance d'entités nommées, d'extraction de relations, d'analyse de sentiment, de résolution d'anaphores et d'extraction de chaînes de coréférence et de regroupement de documents (*clustering*). Nous introduirons à ce niveau une présentation des techniques d'apprentissage automatique (*machine learning*), dont l'importance est grandissante en TAL, et que nous avons mis en œuvre dans certains des composants.

5. Applications

La partie VI (page 131) donne plusieurs exemples d'applications complètes réalisées grâce aux composants d'Antelope. Nous y présentons en premier des applications opérationnelles développées par l'équipe Proxem dans différents domaines : veille économique, e-réputation (analyse d'avis de consommateurs) et ressources humaines. Nous introduisons ensuite une démarche semi-supervisée d'acquisition de connaissances à large échelle. Nous montrons enfin que la plate-forme a aussi été mise en œuvre par plusieurs équipes de recherche, sans nécessiter d'interaction avec Proxem.

6. Interface syntaxe-sémantique

La partie VII (page 153) commence par dresser un premier bilan des objectifs que nous estimons avoir atteints. Elle trace ensuite la route qui reste selon nous à parcourir pour concrétiser la réalisation d'une ISS opérationnelle. Un point critique concerne la désambiguïsation, aussi fine que possible, des différents éléments langagiers. Ce point soulève une question importante : chaque composant d'analyse effectue une tâche particulière ; quand un composant gère une ambiguïté, il porte généralement un jugement dont la portée n'est que locale. Une difficulté consiste donc à tenir compte de l'ensemble des contraintes obtenues localement et à s'assurer de leur cohérence globale, en résolvant les éventuelles contradictions. Comment assurer alors une orchestration d'ensemble qui gère les ambiguïtés avec une vision globale ?

Enfin, nous présentons un prototype d'ISS, qui utilise la sortie de l'ensemble des composants d'analyse syntaxique et sémantique. Cette ISS est définie par des règles de correspondance entre représentation syntaxique et représentation sémantique, ces règles étant obtenues avec une *approche paresseuse*.

7. Conclusion, référence et annexes

La partie VIII (page 171) conclut cette thèse en dressant un bilan des travaux effectués et ouvre des perspectives. Les références incluent la bibliographie (page 173) ainsi que les adresses des sites Web où peuvent être téléchargées les nombreuses ressources citées dans le document (page 184). Enfin,

les annexes contiennent une introduction au Web sémantique (page 187), des précisions sur les éléments mathématiques sous-jacents au regroupement spectral et à l'apprentissage par CRF (page 201), et pour finir des références linguistiques (page 209).

F. Conventions et notations

Nous distinguerons dans ce document les notions de lexie (unité lexicale, association d'un signifiant et d'un signifié) et de vocable (unité polysémique regroupant différentes lexies de même signifiant). Les VOCABLES et LEXIES seront écrits en petites majuscules, les lexies d'un même vocable étant différenciées par des suffixes #1, #2, #3, etc.

Nous ajouterons si besoin après la lexie, en indice entre crochets, une indication permettant au lecteur humain de déterminer son sens. Par exemple, le vocable BAGUETTE a plusieurs sens, correspondant à autant de lexies :

- BAGUETTE#1$_{\text{[pain]}}$ désigne celle fabriquée par le boulanger.
- BAGUETTE#2$_{\text{[instrument de musique]}}$ celle utilisée par le chef d'orchestre.
- BAGUETTE#3$_{\text{[objet magique]}}$ celle du sorcier.

D'une façon générale, nous noterons LEXIE#i la i^{ème} lexie d'un vocable dans notre lexique sémantique (WordNet dans le cas des mots anglais). Si nécessaire, nous préciserons entre apostrophes la traduction en français d'un terme anglais, comme dans l'exemple suivant : MINK#3$_{\text{[animal]}}$ 'vison'.

Nous représenterons un concept (ou *synset* dans le jargon de WordNet) en tant qu'ensemble de lexies synonymes regroupées entre accolades, comme par exemple {NATURAL LANGUAGE#1, TONGUE#2}. Dans les cas où il n'y a pas de risque d'ambiguïté, nous nous autoriserons un raccourci en ne gardant que la première lexie : {NATURAL LANGUAGE#1} ; s'il s'agit du premier sens du vocable, nous pourrons aussi omettre le suffixe de sens : {NATURAL LANGUAGE}.

Les rôles thématiques seront notés en italiques (*Agent*, *Patient*…). Leurs contraintes de sélection seront indiquées entre chevrons (<humain>, <animé>, <comestible+solide>…).

Partie II. Formalismes de représentation du sens d'un énoncé

A. Représentations du sens

1. Qu'est-ce qu'une représentation du sens ?

Définir le sens n'est pas une chose aisée. Pour les linguistes, le sens des réalisations orales (parole) ou écrites (texte) s'appréhende en premier lieu à travers des entités linguistiques factuelles telles que les mots, groupes de mots, phrases… Ces objets perceptibles renvoient à quelque chose de moins perceptible. Dans le texte fondateur de la linguistique moderne, (de Saussure, 1916) estime que « *le signe est un objet à double face* » : il a un côté perceptible (acoustique dans le cas de la parole)[13] et l'autre non perceptible, qui en constitue la face conceptuelle (ou « signifié »).

Le linguiste s'attache donc à définir l'une et l'autre de ces faces. La recherche du sens passe alors par celle des règles selon lesquelles les entités linguistiques de base peuvent se combiner pour permettre à la propriété « avoir du sens » de se transmettre progressivement à des objets de plus en plus complexes. Le principe de compositionnalité, introduit par Frege, consiste à interpréter une expression complexe en fonction de l'interprétation de ses parties et de la manière dont elles sont assemblées.

Les linguistes sémanticiens s'attachent à donner une interprétation extra-linguistique de cette face non perceptible. (Pottier, 1992) présente la sémantique comme une science qui « *se préoccupe des mécanismes et opérations concernant le sens, à travers le fonctionnement des langues naturelles. Elle tente d'expliciter les liens qui existent entre les comportements discursifs baignés dans un environnement toujours renouvelé, et les représentations mentales qui semblent être partagées par les utilisateurs des langues naturelles* ».

Les logiciens (Aristote, Montague…) cherchent à formaliser la « représentation du sens » en construisant un système (métalinguistique, logique, symbolique, mathématique, etc.) permettant de parler du sens linguistique, des connaissances ou des opérations de construction du sens et des connaissances. Apparaissent alors les notions de formalisation, de modélisation et de représentation (avec parfois pour ce dernier terme une confusion entre le mécanisme de production et le résultat produit).

[13] Plus précisément, l'association entre sens et son qui a lieu dans le cerveau ne se fait pas avec le son lui-même, mais avec une représentation de ce son dans le cerveau, que Saussure nomme l'image acoustique : « *[il faut] distinguer les parties physiques (ondes sonores) des physiologiques (phonation et audition) et psychiques (images verbales et concepts). Il est en effet capital de remarquer que l'image verbale ne se confond pas avec le son lui-même et qu'elle est psychique au même titre que le concept qui lui est associé.* » (de Saussure, 1916 : 28-29)

2. Nos objectifs dans ce domaine

Nous ne chercherons ici ni à théoriser sur ce qu'est le sens d'un énoncé (d'autres l'ayant déjà fait avec brio ailleurs), ni à le comprendre *complètement*. Atteindre un niveau fin de compréhension automatique d'un énoncé est un objectif notoirement difficile ; par exemple, la traduction automatique de qualité humaine est réputée impossible dans l'état actuel des connaissances. Cet objectif ambitieux a été régulièrement ajourné au profit de tâches plus locales et moins complexes. Les travaux menés ont néanmoins permis de réaliser des percées concrètes. Ils ont débouché sur quelques applications industrielles, rendant chaque jour un service réel à des dizaines de millions d'utilisateurs : on peut notamment citer la recherche de documents (sur le Web ou dans le cadre d'applications spécialisées dans un domaine) et la correction grammaticale.

Notre objectif est de présenter une démarche outillée pour améliorer les applications analysant de grandes quantités de textes tout-venant[14]. Nous cherchons, modestement, à extraire *une plus grande partie* du sens[15] contenu dans des énoncés. Nous souhaitons contribuer ainsi à améliorer le fonctionnement actuel des applications industrielles de TAL.

3. « Représentation du sens » utilisée par un moteur de recherche

Le moteur de recherche est probablement l'application du TAL la plus populaire à ce jour (en termes de nombre d'utilisateurs) ; c'est aussi la plus aboutie d'un point de vue industriel, en ayant démontré sa capacité[16] à indexer des dizaines de milliards de pages Web. Nous proposons ici de faire un zoom sur les traitements effectués lors de l'indexation d'un texte par un moteur de recherche, et de nous poser la question de *ce que voit la machine*, de sa représentation du sens contenu dans un énoncé.

Notre propos est de mettre en évidence que les opérations successives effectuées aboutissent à une compression destructive[17] de l'information initiale. Nous présenterons juste après la représentation plus riche que nous souhaiterions produire. L'énoncé dont nous allons détailler l'analyse est un avis de consommateur ; voici dans le détail les différentes opérations usuellement effectuées par un moteur de recherche.

a) *Découpage du texte en mots (tokens)*

Partons de l'avis suivant, formulé par un consommateur suite à une visite dans un hypermarché :

je tenais à féliciter la caissiére Céline pour son accueil chaleureux et souriant du samedi 16 février malgré la foule incroyable ce jour la , elle a su faire abstraction de cela et garder le sourire et la bonne humeur . FELICITATIONS

[14] Les documents sur lesquels nous avons concrètement travaillé sont de natures variées : avis de consommateurs, textes encyclopédiques, articles de presse, documents RH. Notons que ces types de textes représentent un pourcentage significatif des documents consultés sur le Web. Ils partagent des propriétés communes (suites d'assertions ou de faits, peu de quantificateurs).

[15] Le périmètre exact de cette *partie* du sens variera en fonction des objectifs d'une application et des contraintes imposées (qualité vs. rapidité du traitement).

[16] Modulo les colossaux investissements d'infrastructure effectués par des géants du Web comme Google, Microsoft et Yahoo. Le parc matériel de Google a été évalué en 2010 à plus d'un million de serveurs.

[17] Une compression est qualifiée de « destructive » quand les données compressées ne permettent pas de reconstruire les données originales.

On constate que ce verbatim contient quelques fautes d'orthographe et une capitalisation abusive ; néanmoins, un lecteur humain n'aura aucune difficulté à le lire, et nul doute sur son intention communicative. D'autres types d'erreurs pourraient rendre la lecture plus difficile pour certaines catégories de lecteurs : abus de style SMS, trop grand nombre d'erreurs de syntaxe, ponctuation totalement absente, etc.

Lors de l'indexation de ce verbatim par un moteur de recherche classique[18], un analyseur de texte commence par découper le texte en *tokens*. Le moteur d'indexation applique ensuite plusieurs filtres consécutivement. Chacun de ces filtres va « simplifier » ou normaliser le texte, au prix d'une perte d'information.

b) Suppression des diacritiques

Le premier filtre consiste généralement à passer tous les mots en minuscule, et à enlever les diacritiques. Le résultat de l'application de ce filtre est alors :

je tenais a feliciter la caissiere celine pour son accueil chaleureux et souriant du samedi 16 fevrier malgre la foule incroyable ce jour la , elle a su faire abstraction de cela et garder le sourire et la bonne humeur . felicitations

L'intérêt de ce premier traitement de surface est de normaliser les termes, et de rendre les recherches subséquentes tolérantes aux petites fautes d'accentuation (huître ≈ huitre, mangé ≈ mange). L'inconvénient est qu'une partie de l'information disparaît : il est difficile de faire ensuite la distinction entre *pâte* et *pâté* par exemple ; de même, des marques telles que *Total* ou *Orange* deviennent indiscernables du nom commun correspondant.

c) Suppression des mots vides

Le filtre suivant enlève notamment les mots grammaticaux (« mots vides » pour reprendre la terminologie de (Tesnière, 1959), *stop words* en anglais). Sont généralement comptés dans cette catégorie les mots qui ne sont pas des noms, verbes, adjectifs et adverbes, ainsi que les formes des auxiliaires *être* et *avoir*. Nous obtenons alors :

je tenais a feliciter la caissiere celine pour son accueil chaleureux et souriant du samedi 16 fevrier malgre la foule incroyable ce jour la , elle a su faire abstraction de cela et garder le sourire et la bonne humeur . felicitations

L'objectif de ce filtre est de diminuer le nombre de termes à indexer, les mots vides étant très fréquents et tellement communs qu'il semble au premier abord inutile de les indexer. Son défaut est de supprimer plusieurs catégories de mots porteurs de sens.

En filtrant aussi les prépositions, le moteur de recherche perd la capacité à établir le contraste entre des livres écrits *pour* des enfants et des livres écrits *par* des enfants ; de même, « Jean dort *chez* Marie » n'a pas exactement le même sens que « Jean dort *avec* Marie ».

Sans les négations, il devient difficile de faire du calcul d'opinions ou de sentiments. Les ponctuations sont aussi généralement enlevées à ce stade ; là aussi, une information utile à l'analyse de sentiments est perdue (par exemple, les émoticônes ou points d'exclamation).

[18] Nous montrons ici les calculs effectués par l'analyseur de Lucene (la référence *open source* dans ce domaine).

Certains adjectifs possessifs sont également intéressants pour désambiguïser le nom qu'ils qualifient. Dans un avis de consommateur, une heuristique simple pour séparer les sens juridique et fruit du nom *avocat* est de regarder le premier mot à gauche (« mon avocat » désignant sans ambiguïté l'homme de loi).

d) Lexémisation

Les mots sont ensuite tronçonnés pour n'en retenir que la racine[19]. Sur notre verbatim, cela donne :

> je **ten**ais a **felicit**er la **caiss**iere **celin**e pour son **accueil** **chaleur**eux et **souri**ant du **samed**i 16 **fevri**er malgre la **foul**e **incroy**able ce **jour** la , elle a **su** **fair**e **abstract**ion de cela et **gard**er le **souri**re et la **bon**ne **humeur** . **felicit**ations

Le principal intérêt de la lexémisation des termes est de faciliter une recherche subséquente sur les concepts approchants, sans avoir à gérer explicitement des relations de dérivation morphologique ; par exemple, une recherche sur « *la* ***chine*** ***investi**t en* ***Afriq**ue* » renverrait aussi un texte parlant d'***investi**ssements* ***chin**ois sur le continent* ***afric**ain*.

La limite de cette approche est d'entraîner souvent une confusion des concepts : en effet, si on peut considérer que ***cheval*** et ***cheval**ier* ont effectivement un rapport, le lien avec ***cheval**eresque* est plus ténu, et celui avec ***cheval**et* est inexistant ; a contrario, la relation avec ***caval**ier* est, quant à elle, purement et simplement ignorée par les algorithmes usuels de lexémisation.

e) Obtention d'un vecteur termes-fréquences

Au final, le verbatim d'origine est transformé en un vecteur associant à la racine de chaque terme sa fréquence dans le document : `{ abstract=1, accueil=1, bon=1, caiss=1, celin=1, chaleur=1, fair=1, felicit=2, fevri=1, foul=1, gard=1, humeur=1, incroy=1, jour=1, samed=1, souri=2, su=1, ten=1 }`. Le sens d'un document est donc représenté par un point dans un espace vectoriel de grande taille, dont les dimensions sont les termes.

Lors d'une recherche, la requête effectuée par un utilisateur est transformée de la même façon sous forme de vecteur normalisé[20] ; le moteur recherche alors au sein de l'index les documents les plus proches, c'est-à-dire ceux maximisant le produit scalaire entre le vecteur de la requête et celui des documents (avec généralement une pondération de type TF-IDF[21] tenant compte de la fréquence des termes au sein de l'index).

f) Bilan

On constate donc que la représentation du sens d'un énoncé par un moteur de recherche sous forme de « sac de mots » compresse l'information d'une façon destructive. On retrouve ici le phénomène bien connu en traitement d'image : une compression excessive entraîne une pixellisation de l'image,

---

[19] La lexémisation (ou *stemming* en anglais) revient à prendre la racine des mots privés de leur terminaison. On obtient après lexémisation d'un mot son lexème, concept souvent synonyme de radical du mot. Dans les exemples que nous donnons ici, l'algorithme de (Porter, 1980) est utilisé.

[20] C'est-à-dire une position sur une sphère de dimension égale au nombre de termes possibles.

[21] TF-IDF (*term frequency-inverse document frequency*) est une méthode de pondération souvent utilisée dans la fouille de textes. Cette mesure statistique nous permet ici d'évaluer l'importance d'un mot au sein d'une définition. Le poids augmente proportionnellement en fonction du nombre d'occurrences du mot dans la définition. Il varie également en fonction de la fréquence du mot dans le corpus formé par l'ensemble des définitions.

et il devient alors difficile de deviner ce qu'elle représente. Pour s'en convaincre, partons du vecteur termes-fréquences `{ exploit=1, system=1 }` ; il peut représenter différents fragments de texte : « *les exploiteurs du système* », « *le système d'exploitation* », « *des exploits systématiques* », « *les exploitants de ce système* »…

Notre propos n'est évidemment pas de critiquer les moteurs de recherche, qui rendent un service indiscutable. Nous tenions juste à souligner que dans la catégorie la plus utilisée des applications de TAL, il n'y a pas, de nos jours, de réelle compréhension d'un document. Pour employer une image, de notre point de vue, les applications industrielles de TAL sont actuellement myopes, borgnes et daltoniennes.

4. Vers une amélioration de cette représentation

La question qui se pose donc est : comment obtenir une représentation *plus fine* du sens dans le cadre d'applications devant traiter rapidement de grandes quantités de textes tout-venant ? Nous souhaitons permettre, par exemple, d'effectuer certains calculs non triviaux :

- Détecter ou produire une paraphrase, c'est-à-dire une phrase ayant le même sens que l'énoncé de référence[22].
- Répondre à une question (dont la réponse est dans le texte) ou faire une inférence (c'est-à-dire répondre à une question dont la réponse n'est pas explicite dans le texte),
- Détecter une opinion positive ou négative sur un fait (« *il faut embaucher cette personne* », « *surtout ne pas aller voir ce film* »…). L'analyse de sentiments est typiquement un calcul de haut niveau, qui nécessite de traiter préalablement plusieurs autres tâches (extraction d'entités et de relations, éventuellement analyse syntaxique).

A partir du début du verbatim précédent, nous souhaiterions obtenir une représentation sémantique ressemblant à la figure 1 (le formalisme exact importe peu ici) :

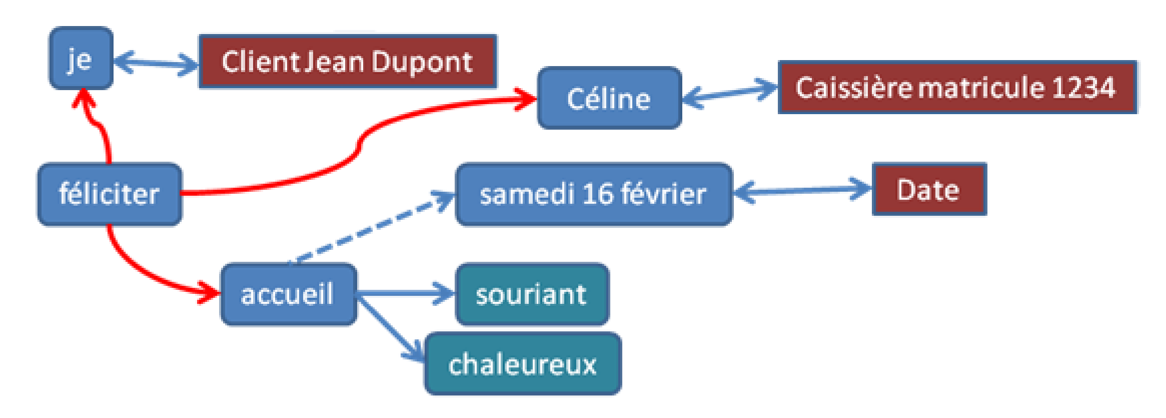


Figure 1 : Représentation sémantique partielle d'un avis de consommateur

Dans cette représentation structurée, l'auteur du verbatim (« je ») serait identifié (par une mise en relation avec l'application de gestion de la relation client), ainsi que la caissière Céline (matricule 1234 dans l'application de gestion des ressources humaines) ; l'expression temporelle serait reconnue en tant que telle, et la date précise calculée ; l'accueil de la caissière qualifié de deux attributs à connotation positive, etc.

[22] On peut considérer que c'est l'un des objectifs de la Théorie Sens-Texte, voir (Milićević, 2007).

B. La Théorie Sens-Texte

Le cadre théorique sur lequel nous nous sommes appuyés pour la conception de la plate-forme est la *Meaning-Text Theory*, ou Théorie Sens-Texte (Mel'čuk, 1988a)[23]. Cette théorie suit la partition classique de la modélisation d'un énoncé en niveaux de représentation phonologique/phonétique, morphologique, syntaxique et sémantique. La spécificité de l'approche Sens-Texte consiste en une subdivision (profond *vs.* de surface) des trois premiers niveaux. Plusieurs implémentations en ont déjà été effectuées[24].

1. Différents niveaux de représentation

Notre objectif est de permettre une analyse de textes qui puisse aller jusqu'à une représentation sémantique, sans que ce soit systématiquement une obligation. On peut imaginer, par exemple, qu'un article d'encyclopédie soit analysé finement sur les premiers paragraphes, et plus superficiellement sur la suite. Nous souhaitons donc représenter différents niveaux d'information au sein d'un même document. Sans forcément viser une implémentation complète et orthodoxe de ce modèle[25], nous mettons en œuvre les niveaux de représentation suivants :

- Morphologie de surface (nous utiliserons dans la suite l'abréviation RMorphS) et morphologie profonde (RMorphP) : ces niveaux permettent de représenter une information linéaire résultant d'un étiquetage morphosyntaxique ou d'un *chunking*.
- Syntaxe de surface (RSyntS) : cette représentation consiste en un arbre de dépendances syntaxiques, dont les nœuds représentent des lexèmes (pleins ou vides) et les arcs des dépendances syntaxiques de surface, spécifiques à une langue donnée (voire à un analyseur syntaxique particulier) ; pour des raisons de commodité, Antelope permet aussi de la représenter sous forme d'un arbre de constituants.
- Syntaxe profonde (RSyntP) : c'est un arbre de dépendances non linéairement ordonné, dont les nœuds sont des unités lexicales et les arcs des dépendances syntaxiques profondes (universelles) ; les unités lexicales sont désambiguïsées et peuvent être des locutions (des expressions multi-mots) ; un lexème vide, comme une préposition régime, n'apparaît pas dans la RSyntP.
- Sémantique (RSém) : c'est un graphe dont les nœuds représentent les sens désambiguïsés des unités lexicales et grammaticales ; les arcs sont des relations prédicat-argument (une représentation équivalente peut être donnée dans le formalisme du calcul des prédicats).

La TST propose un modèle bidirectionnel, utilisable en analyse ou en génération. Notre objectif vise, dans le cadre de nos travaux actuels, à extraire des connaissances d'un texte ; nous mettons donc en œuvre des interfaces unidirectionnelles réalisant les transitions Texte ⇒ RMorphS ⇒ RMorphP ⇒ RSyntS ⇒ RSyntP ⇒ RSém. Notre contribution spécifique porte sur l'ISS, qui effectue les transitions RSyntS ⇒ RSyntP et RSyntP ⇒ RSém, le passage d'un niveau au suivant étant effectué par une interface clairement définie.

---

[23] Pour une modélisation formelle de la TST, on pourra consulter (Kahane, Mel'čuk, 1999), ainsi que (Kahane, 2002) pour une grammaire d'unification basée sur la TST.

[24] Notamment en milieu industriel, parfois en simplifiant le modèle ; citons par exemple la Cogentex (Iordanskaja, Kittredge, Polguère, 1998 ; Lavoie et al., 2000), Lexiquest (Coch, 1998) et VirtuOz.

[25] Nous n'avons, par exemple, pas de niveau phonologique, ni de distinction thème/rhème.

2. Composants effectuant les transitions entre niveaux de représentation

Atteindre une représentation sémantique nécessite la mise en œuvre de plusieurs types de traitements complémentaires. L'implémentation des *transitions* entre niveaux de représentation se fait donc grâce à différents types de composants, comme précisé dans le tableau 1. On y remarquera que certains d'entre eux (résolution d'anaphores, désambiguïsation lexicale) peuvent se retrouver dans plusieurs transitions ; en effet, leur traitement sait s'adapter en fonction du niveau où ils s'appliquent, en donnant un résultat plus ou moins précis.

Le fonctionnement exact des composants utilisés dans Antelope est détaillé au chapitre III.I (pour les tâches allant du prétraitement jusqu'à l'analyse syntaxique) et dans la partie V (pour les traitements sémantiques). Certains ont été réalisés spécifiquement pour la plate-forme, d'autres ont une origine extérieure[26].

Transition	Type de composant de traitement mis en œuvre
Texte ⇒ RMorphS	Accès au lexique Segmentation (texte brut ou document HTML) Étiquetage morphosyntaxique Identification des expressions multi-mots Désambiguïsation lexicale (basique) Résolution d'anaphores et de coréférences (basique)
RMorphS ⇒ RMorphP	*Chunking*
RMorphP ⇒ RSyntS	Analyse syntaxique par dépendances ou en constituants Désambiguïsation syntaxique Désambiguïsation lexicale (intermédiaire)
RSyntS ⇒ RSyntP	Analyse syntaxique profonde Résolution d'anaphores et de coréférences (intermédiaire)
RSyntP ⇒ RSém	Étiquetage des rôles sémantiques Désambiguïsation lexicale (avancée) Résolution d'anaphores et de coréférences (avancée)

Tableau 1 : Composants typiquement utilisés pour implémenter une transition

C. Notre représentation sémantique idéale

Nous prendrons comme exemple la phrase suivante, tirée de l'article *Amazon River* de l'encyclopédie Britannica : « *The first European descent was made by Fransesco de Orellana in 1541* ». La figure 2 présente la représentation sémantique que nous aimerions idéalement être capables de calculer.

Cette représentation fait référence à différentes ressources lexicales (WordNet, VerbNet, The Preposition Project, NomLex). Nous les présenterons plus en détail dans la partie IV.

[26] La section III.H.3 précise la façon dont les composants externes sont intégrés.

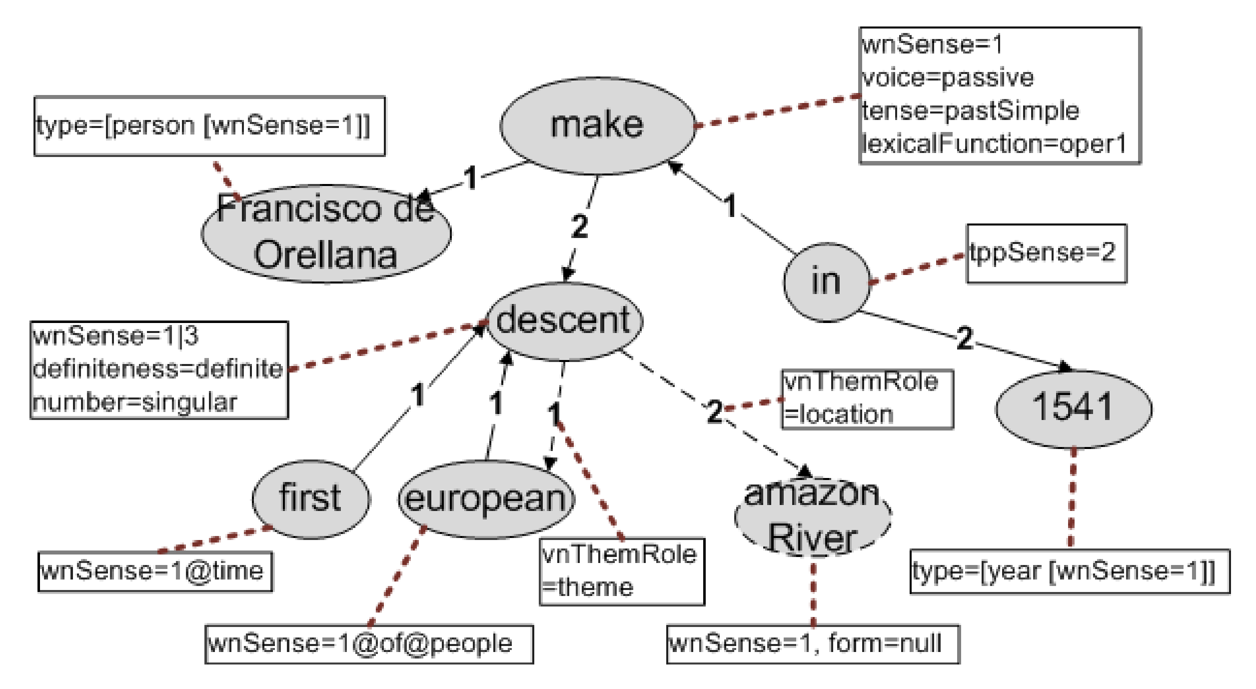


Figure 2 : Représentation sémantique « idéale » que nous souhaitons obtenir

Dans cette représentation, les unités lexicales de type nom, verbe, adjectif ou adverbe, sont désambiguïsées par rapport à WordNet, avec un trait [WNSENSE]. Par exemple, DESCENT [WNSENSE=1|3] indique que WordNet propose deux sens du nom DESCENT compatibles avec le contexte de la phrase (même si aucun n'est parfaitement adapté ici) : DESCENT#1 (*a movement downward*) et DESCENT#3 (*the act of changing your location in a downward direction*).

Pour les prépositions, notre ressource de référence est *The Preposition Project* (Litkowski, 2002). L'annotation IN [TPPSENSE=2]) indique qu'il s'agit ici du second sens de la préposition IN, définie par : IN#2 (*expressing a period of time during which an event happens or a situation remains the case*).

Lorsqu'une définition est trop générale, nous pouvons même pointer vers une composante de cette définition[27] comme dans FIRST [WNSENSE=1@TIME] ou EUROPEAN [WNSENSE=1@OF@PEOPLE]. Une ressource de type DiCouèbe nous permettrait d'indiquer que MAKE est ici la valeur de la fonction lexicale Oper1 (MAKE [LEXICALFUNCTION=OPER1]), c'est-à-dire un verbe support vide[28].

Certains éléments lexicaux ne sont pas dans notre lexique, mais peuvent être reconnus comme des entités nommées dont on indiquera le type, qui est lui-même une unité lexicale avec sa référence : FRANCISCODEORELLANA [TYPE=[PERSON [WNSENSE=1]]] ; 1541 [TYPE=[YEAR [WNSENSE=1]]]. Enfin, étant donné qu'on est dans l'article de l'*Amazon River*, on peut lever une anaphore zéro sur le deuxième argument de *descent*. Le trait [FORM=NULL] indique que le nœud sémantique AMAZONRIVER n'est pas lexicalement réalisé.

---

[27] WordNet propose un seul sens pour l'adjectif *European*, mais celui-ci est doublement disjonctif, ce qui donne en tout 6 sens : EUROPEAN#1 -- *of or relating to or characteristic of Europe or the people of Europe*. Le sens proposé pour *first* est également disjonctif : FIRST#1 -- *preceding all others in time or space or degree*. Cf. l'ambiguïté de *Microsoft est le premier producteur de logiciels.*

[28] Cela n'implique pas que MAKE#1 n'est pas un verbe plein, mais que ce verbe n'a pas d'apport sémantique supplémentaire par rapport au nom prédicatif qu'il supporte, ici *descent*. L'information qu'il s'agit d'un verbe support vide est importante, puisqu'elle signifie que le verbe a essentiellement un rôle syntaxique et qu'il peut ne pas avoir de correspondant dans une paraphrase, comme *The Amazon River was descended for the first time by an European in 1541 by Francisco de Orellana*.

Les traits grammaticaux sont également calculés, comme le temps pour un verbe ou la définitude et le nombre pour un nom (DESCENT [DEFINITENESS=DEFINITE, NUMBER=SINGULAR]). Dans l'absolu, les temps verbaux devraient être désambiguïsés, mais nous ne connaissons pas de ressource qui nous propose des valeurs. Nous indiquons également la voix, qui, bien que n'ayant généralement pas d'incidence sur le contenu propositionnel, joue un rôle dans la structure communicative.

Les différents arguments de chaque unité lexicale sont numérotés arg1, arg2, etc. Lorsqu'il s'agit de verbes dont le cadre de sous-catégorisation est décrit dans une ressource comme VerbNet, on peut préciser un rôle thématique. On notera que le verbe MAKE n'est pas dans cette ressource. Par contre, la ressource NomLex décrit précisément la relation de dérivation[29] entre le nom DESCENT et le verbe DESCEND, permettant de déduire que *"descent [of the Amazon River] by Francisco_de_Orellana"* est équivalent à *"Francisco_de_Orellana descends [the Amazon River]"*. WordNet décrit également une relation de dérivation morphologique entre le nom DESCENT#1 et le verbe DESCEND#1. WordNet indique juste que ce verbe est intransitif (*Someone/something ~s*) ; VerbNet en décrit aussi l'usage transitif[30], où l'on récupère les rôles thématiques ARG1[VNTHEMATICROLE = *Theme*] et ARG2[VNTHEMATICROLE = *Location*].

Nous conservons dans notre représentation sémantique une trace de la structure hiérarchique de la syntaxe. Lorsqu'une relation prédicat-argument correspond à une dépendance syntaxique dans le même sens, nous disons que celle-ci est directe, et sinon qu'elle est inverse. Les relations inverses sont celles entre un modifieur et son gouverneur syntaxique (de DESCENT vers EUROPEAN, sur la figure 2). Les relations prédicat-argument qui ne correspondent pas directement à une relation syntaxique sont dites virtuelles (de DESCENT vers AMAZONRIVER sur la figure 2). Cette structure hiérarchique nous permet de contrôler la portée de certains éléments. Ainsi FRANCISCO DE ORELLANA n'est pas dans la portée de FIRST et notre phrase ne pourrait être correctement paraphrasée par *The first descent of Francisco de Orellana was made in 1541* ou *The first descent in 1541 was made by Francisco de Orellana*.

Notre représentation sémantique est directement inspirée des représentations sémantique et syntaxique profonde de la Théorie Sens-Texte (Mel'čuk, 1988a ; Candito, Kahane, 1998 ; Kahane, 2002) et adaptée en fonction des ressources dont nous disposons. Des représentations similaires ont été proposées par d'autres auteurs, sans référence explicite à la Théorie Sens-Texte[31].

Il s'agit d'une représentation sémantique du contenu linguistique et non d'une sémantique dénotationnelle comme les représentations sémantiques basées sur la logique. Il n'y a donc pas à proprement parler de calcul de valeurs de vérité associées. Par contre, ce type de représentation permet des calculs de paraphrases (Mel'čuk, 1988b ; Milićević, 2007) et a été implémenté avec succès pour la génération de textes (Iordanskaja *et al.*, 1988 ; Bohnet, Wanner, 2001) ou la traduction automatique (Apresjan *et al.*, 2003). De telles représentations permettent également de faire de l'extraction d'information (par unification partielles de structure et applications de règles de paraphrasage) et de répondre à des questions telles que *Quelle a été la première personne à descendre l'Amazone ?* (Chaumartin, 2007b), détaillé ici en page 68, décrit une stratégie d'extraction

---

[29] NomLex donne une correspondance précise entre les arguments du nom et ceux du verbe dont il dérive : NOM :ORTH "descent" :VERB "descend" :NOM-TYPE ((VERB-NOM)) :VERB-SUBJ ((DET-POSS) (PP :PVAL ("by"))).

[30] Dans le cadre de sous-catégorisation ESCAPE-51.1-1.

[31] Voir par exemple DMRS (Copestake, 2009), (Bédaride, Gardent, 2009) et (Bonfante, Guillaume, Morey, Perrier, 2010).

automatique de règles de paraphrase par alignement d'articles d'encyclopédies utilisant des représentations de ce type.

Comme notre description des traits figurant dans la représentation sémantique a pu le montrer, le calcul d'une telle représentation met en jeu de nombreuses ressources lexicales (que nous présentons en partie IV) et plusieurs types de calculs (détaillés en partie V) : désambiguïsation lexicale, reconnaissance et typage d'entités nommées, résolution d'anaphores, etc. Nous présentons en section VII.C les règles de correspondance proprement dites qui permettent de construire le graphe sémantique, de lever certaines ambiguïtés et de reconstruire certaines dépendances virtuelles.

D. Autres formalismes de représentation du sens

La TST n'est pas le seul formalisme qui propose une représentation du sens et ayant débouché sur plusieurs implémentations effectives. Nous pouvons citer en particulier les graphes conceptuels (GCs), initialement introduits dans (Sowa, 1976) et dont on peut trouver une présentation plus récente dans (Nazarenko, 2004).

Les graphes conceptuels sont un système logique inspirés des graphes existentiels de Charles Sanders Peirce et des réseaux sémantiques utilisés en intelligence artificielle. Leur intérêt est de permettre la représentation du sens sous une forme précise (du point de vue de la logique), d'être faciles à lire pour un humain, et d'une complexité suffisamment raisonnable pour que des systèmes informatiques puissent effectuer des calculs dessus. Les graphes conceptuels peuvent servir de langage intermédiaire entre des formalismes informatiques et des langues naturelles, dans un sens comme dans l'autre. Les GCs ont été mis en œuvre dans le domaine du TAL (notamment en recherche d'information) mais aussi pour la conception de bases de données et le développement de systèmes experts. La figure 3 montre le GC représentant la phrase « *John is going to Boston by bus* ».

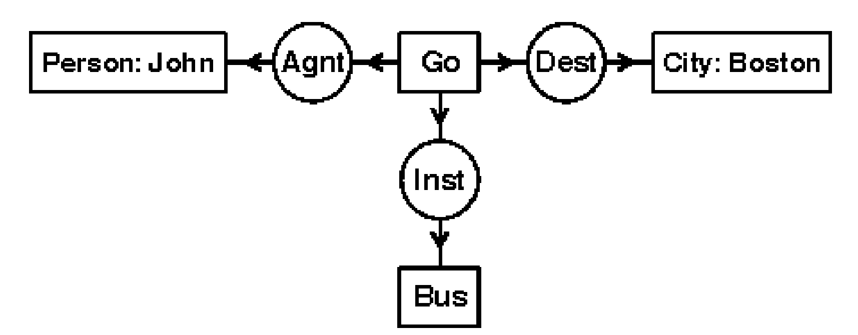


Figure 3 : Représentation d'une phrase simple avec le formalisme des graphes conceptuels

Notons que le formalisme autorise aussi une forme linéaire d'écriture, avec un mécanisme de coréférence. Le même GC peut s'écrire :

```
[Go]-
   (Agnt)->[Person: John]
   (Dest)->[City: Boston]
   (Inst)->[Bus].
```

Partie III. Antelope : une plate-forme pour extraire les sens du texte

A. Objectifs de la plate-forme

Nous avons présenté jusqu'ici nos objectifs, les principes généraux de la plate-forme de TAL que nous avons réalisée, et les formalismes de représentation du sens d'un énoncé que nous cherchons à obtenir. Cette partie va rentrer dans le détail de notre plate-forme, baptisée Antelope (*Advanced Natural Language Object-oriented Processing Environment*), présenter sa conception et la comparer à d'autres projets de référence. Elle souligne aussi les précautions architecturales à prendre pour qu'un tel développement complexe reste maintenable.

En partie basée sur la Théorie Sens-Texte, Antelope permet l'analyse syntaxique et sémantique de textes sur des corpus de volume important. Un effort d'intégration, reposant sur des « bonnes pratiques » de génie logiciel, permet de rendre interchangeables les différentes ressources dédiées à une même tâche. Par exemple, l'étiquetage morphosyntaxique dans une langue peut être effectué par plusieurs composants. Le choix du meilleur d'entre eux dépend typiquement du corpus à traiter ; une application peut alors proposer à l'utilisateur de choisir le composant le mieux adapté dans un contexte donné.

Antelope intègre plusieurs composants préexistants, notamment pour l'analyse syntaxique. Notre contribution directe concerne la constitution d'un lexique sémantique à partir de données linguistiques à large couverture provenant de différentes sources (voir la partie IV, page 41), l'ajout de composants d'analyse sémantique (décrits plus précisément en partie V, page 89) et la formalisation d'un modèle linguistique unifié (présenté au chapitre C, page 22).

Antelope propose une chaîne complète de traitement du langage. Conçue initialement pour l'anglais, pour des raisons de disponibilité de ressources dans cette langue, la plate-forme a été ensuite adaptée au français. Elle est progressivement enrichie pour traiter d'autres langues européennes. La prise en compte du multilinguisme dans la plate-forme est détaillée au chapitre D (page 24).

Antelope vise à être simple à mettre en œuvre, pour en permettre l'utilisation par un informaticien n'ayant pas de connaissances particulières en linguistique[32]. Pour cela, les principaux composants disposent de paramétrages par défaut, privilégiant un mode de traitement (rapide ou précis). Un utilisateur expert aura en revanche la possibilité de jouer sur des paramètres plus fins, ou d'inclure ses propres modules dédiés à une tâche donnée. Le niveau de traitement le plus complet vise à extraire d'un texte tout un ensemble de connaissances sémantiques (entités nommées, relations, coréférences, multiples sens des mots…), et de les représenter à l'aide du formalisme décrit au chapitre II.C ; on peut choisir de préserver les ambiguïtés, ou de ne conserver que le meilleur sens

[32] On remarquera que les outils de TAL ont plus souvent l'objectif inverse.

calculé. Cette souplesse permet donc d'extraire différents niveaux de sens à partir d'un même texte, et de choisir celui qui est pertinent à calculer dans un contexte applicatif donné.

B. Diversité des éléments à analyser

Commençons par détailler les tâches élémentaires qu'un programme informatique peut effectuer et les informations qu'il peut associer, lors de l'analyse d'un ensemble de textes, à chacun de ses éléments : mot, phrase, paragraphe, document, en allant jusqu'au corpus dans sa globalité.

1. Mot

Le mot est l'élément atomique constituant un texte. Plusieurs opérations sont possibles sur un mot : correction orthographique, calcul de sa langue, de sa racine, d'une étiquette morphosyntaxique, de sa flexion, de sa forme de base, de sa valence…

Énumérer les sens possibles d'un mot est un problème discret[33] qui se résout par rapport à un lexique de référence. La notion de lexique apparaît donc dès l'analyse des constituants les plus fins des textes ; nous la préciserons dans la partie IV, consacrée aux données linguistiques.

2. Phrase

L'analyse de la phrase vise à déterminer les relations que les mots entretiennent entre eux. Des ambiguïtés existent sur ces relations, par exemple sur les rattachements prépositionnels. Plusieurs modèles de représentation sont possibles, par exemple sous forme d'arbres de constituants ou d'arbres de dépendances syntaxiques.

Plusieurs mots peuvent se regrouper au sein d'expressions plus ou moins figées. Certaines langues compositionnelles forgent des mots composés complexes (par exemple, en allemand, *Donaudampfschifffahrtsgesellschaft* = Société de navigation à vapeur du Danube).

Plusieurs prédicats peuvent être énoncés dans une même phrase, reliés entre eux par des relations du discours ou des conjonctions. Une prédication peut être exprimée avec des mots ordonnés selon un ordre canonique (sujet, verbe, complément). Toutefois, les textes sont rarement écrits aussi simplement : la présence de relatives, de constructions passives, de verbes à montée ou à contrôle… permettent des constructions arbitrairement complexes.

La présence de plusieurs mots dans la phrase (et dans les phrases voisines) crée un contexte qui aide à identifier le sens d'un mot parmi ceux qui sont possibles. Des algorithmes de désambiguïsation peuvent alors exploiter les différents indices présents.

Une phrase peut se représenter sous forme de graphe syntaxique (de surface ou profond) ou de graphe sémantique, en fonction de la précision de l'analyse qu'on souhaite effectuer.

[33] Problème discret, du moins en TAL, où la désambiguïsation revient à choisir *l'un* des sens parmi ceux proposés sous forme d'une liste *finie* dans un lexique de référence. La polysémie semble être un problème universel, dans la mesure où ce phénomène se retrouve dans toutes les langues, et concerne en premier des mots du quotidien. (Victorri, Fuchs, 1996) propose, dans le cadre d'une conception dynamique et continue de la construction du sens, une explication des mécanismes cognitifs permettant à un locuteur humain de traiter avec la même facilité les mots polysémiques et les mots monosémiques.

3. Document

La représentation d'un document étend et généralise celle de la phrase. Les anaphores intra-phrastiques étant relativement rares[34], elles ne sont généralement pas explicitées dans les modèles de représentation de la phrase. Au niveau du document, il devient indispensable de prendre en compte les informations concernant les anaphores (pronominales, nominales, événementielles…). Les composantes connexes du graphe des anaphores constituent des chaînes de coréférence concernant une entité ou un événement particulier. D'autres types d'extractions de connaissances sont envisageables au niveau du document : résumé, opinions, thématiques…

4. Paragraphe

Segmenter un document en phrases et en mots n'est pas une opération triviale (Grefenstette, 1994). Un niveau intermédiaire entre la phrase et le document est le paragraphe, un segment de texte compris entre deux alinéas. Le paragraphe est un élément de structure essentiel du document : un changement de paragraphe marque la composition du texte en termes d'interlocution ou le désir de l'auteur de mettre en avant un énoncé ou une idée.

Il peut s'agir d'un titre ou d'un élément d'énumération (un style est alors associé au paragraphe) ou d'un ensemble de phrases ; dans le premier cas, la présence d'un style permet de connaître l'importance du paragraphe ; dans le second cas, la présence d'un nombre suffisant de mots permet de calculer sa thématique.

Les paragraphes sont eux-mêmes organisés au sein d'éléments de structure, tels que des sections ou parties d'un document.

5. Corpus

Disposer d'un grand nombre de documents dans un domaine donné, partageant un certain degré d'homogénéité, offre un volume « suffisant » de texte qui permet de tirer parti des hypothèses distributionnelles (loi de Zipf et de Harris). L'exploitation endogène d'un corpus permet d'extraire les principaux termes du domaine (en effectuant une extraction terminologique), d'effectuer des opérations d'apprentissage automatique (classification…) ou encore de regrouper automatiquement des documents en sous-ensembles thématiques (appelés *clusters* en anglais).

6. Types de corpus analysés avec Antelope

Antelope a déjà été concrètement utilisée pour analyser des documents issus de corpus de natures variées, en anglais et en français, et de qualité rédactionnelle variable. Nous présentons ici rapidement la nature de ces corpus.

a) Articles encyclopédiques

Nous avions initialement concentré nos travaux sur des articles encyclopédiques ; ces documents sont en effet généralement bien écrits et factuels, et se prêtent donc bien à un traitement automatique. Nous avons notamment effectué des analyses sur la Wikipédia française et sur la *Simple Wikipedia*. Cette dernière est particulièrement destinée aux enfants anglophones et aux adultes dont l'anglais n'est pas la langue maternelle ; elle est écrite avec une grammaire et un champ

[34] Le lecteur attentif aura néanmoins remarqué que cette phrase et la précédente en contiennent…

lexical simplifiés, ce qui en rend la lecture en principe plus simple pour le public visé. Nous formulons l'hypothèse que cette encyclopédie a une caractéristique intéressante pour le TAL : si est elle plus simple à lire pour des humains (par rapport à l'English Wikipedia complète), elle devrait être aussi plus facile à traiter pour un analyseur syntaxique. D'autre part, elle compte moins d'articles, ce qui autorise une analyse d'ensemble plus rapide[35].

b) Articles de presse

Antelope a été utilisée dans le cadre du projet SCRIBO, présenté en détail au chapitre VI.A, page 132. L'un des objectifs de ce projet est l'extraction d'information (personnes, lieux, organisations) à partir d'articles de presse en français ou en anglais, émanant notamment de l'Agence France-Presse. SCRIBO utilise des annotateurs en architecture UIMA, et représente les informations avec des standards du Web sémantique.

c) Avis de consommateurs

Antelope est le moteur d'Ubiq, une solution de gestion d'e-réputation présentée au chapitre VI.D, page 141. Ubiq extrait des informations à partir d'avis de consommateurs, et détermine ce qui se dit autour d'une marque donnée et de ses concurrentes, pour répondre aux questions suivantes : quels sont les sujets dont parlent les consommateurs, de quoi sont-ils satisfaits ou mécontents, et quelles sont leurs attentes ? En regroupant les informations d'une même période temporelle, Ubiq détecte aussi les tendances, et permet d'anticiper des alertes telles que des risques sanitaires ou juridiques.

Ces avis sont collectés soit à partir du Web public (notamment de blogs et de forums), soit à partir d'emails envoyés spontanément à la marque ou de retranscription de conversation téléphonique. Ces documents sont donc parfois très mal écrits, et font l'objet d'une correction orthographique avant analyse.

d) Offres d'emplois et CV

Antelope a aussi été utilisée avec succès pour l'analyse de documents dans le domaine des Ressources Humaines, offre d'embauche ou *Curriculum Vitae* de candidat. L'adaptation d'Ubiq au domaine RH est présentée au chapitre VI.E, page 146.

Antelope permet une extraction fine des postes, compétences, talents, diplômes, lieux, langues, etc. Les résultats obtenus autorisent ensuite à trouver les offres d'emploi correspondant le mieux à un CV donné ou, d'une façon symétrique, de filtrer les profils de candidat pour retenir ceux qui sont les mieux adaptés à une offre.

C. Modèle unifié des niveaux de représentation linguistique

1. Conception des échanges dans la plate-forme

Une approche répandue dans la fabrication des chaînes de traitement en TAL consiste à exécuter séquentiellement plusieurs programmes, chacun d'entre eux se focalisant sur une tâche particulière. Les données échangées sont spécifiées *via* un format d'entrée et un format de sortie attendus par

[35] 15 000 articles en 2007 lors de nos premiers travaux dessus ; 72 000 articles en juin 2011 (soit 50 fois moins d'articles que la version anglaise complète, qui en compte 3 650 000 à la même date).

chaque programme. Un tel pipeline de traitement représente l'architecture la plus élémentaire d'un système modulaire. Ce modèle nous semble toutefois comporter deux faiblesses. En premier lieu, le lancement d'un nouvel exécutable a un coût fixe en ressources, ce qui rendrait pénalisante la multiplication de tâches courtes (portant sur un mot isolé, par exemple). En second, un programme donné va se spécialiser sur une tâche particulière ; il est toujours possible d'élargir le champ de son traitement, mais cela se fait en complexifiant les formats d'entrée et de sortie.

Pour faciliter l'intégration dans Antelope de différents types de composants de traitement, nous avons choisi une approche plus complexe que celle du pipeline, de façon à favoriser la coopération entre composants de traitement linguistique. Un modèle de données unifié regroupe les différents niveaux de représentation de la TST. Les composants de traitement peuvent alors interagir facilement avec ces données.

2. Présentation du modèle de données unifié

Tous les objets linguistiques résultant de l'application d'un traitement (phrase, mot, dépendance, sens de mot, rôle syntaxique, rôle sémantique, anaphore…) sont définis dans un modèle de données unifié, décrit à la figure 4. Ce modèle correspond à l'union de l'ensemble des formats d'entrée et de sortie des composants utilisés.

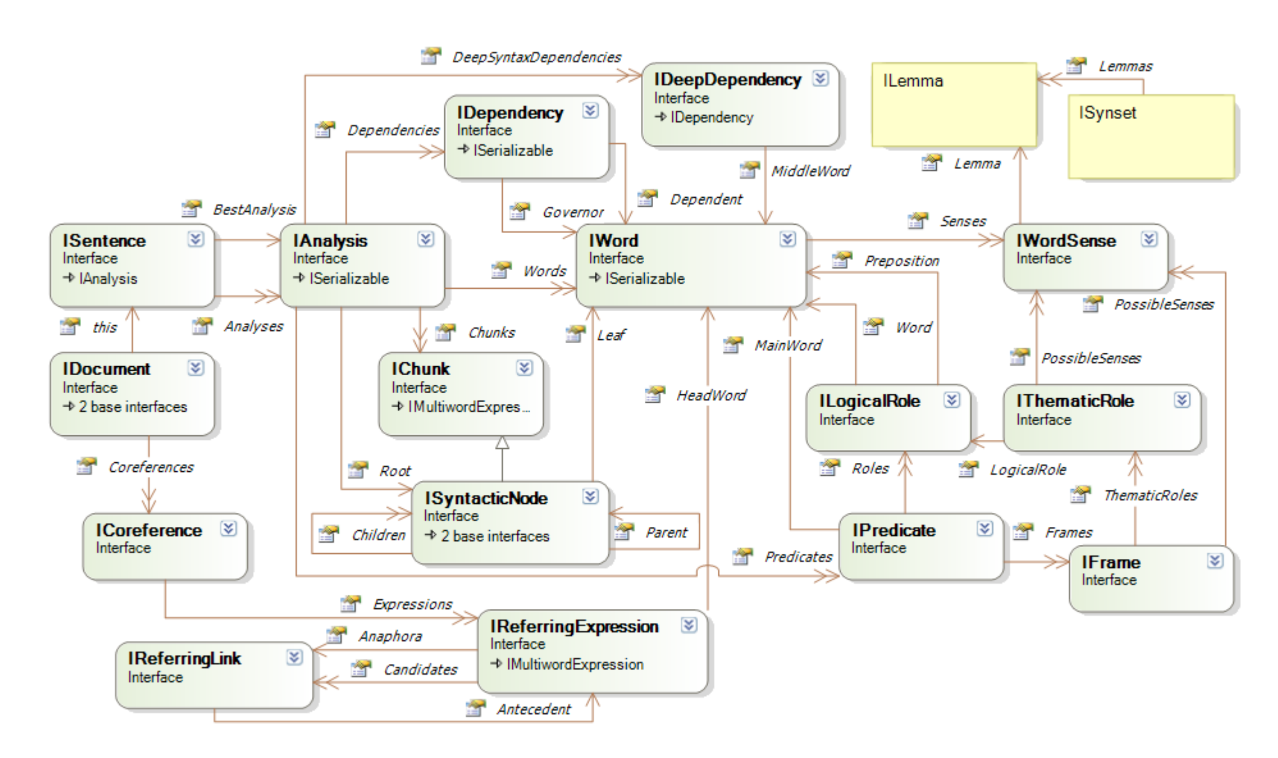


Figure 4 : Vue d'ensemble du modèle de données linguistiques unifié défini pour Antelope

Le formalisme graphique utilisé ici est proche d'UML. Un rectangle arrondi représente une classe[36]. Les relations entre classes sont matérialisées par des flèches ; une flèche simple représente une référence vers un seul objet (ou une référence nulle) ; une flèche double symbolise un lien vers une liste d'objets. Pour des raisons d'espace, les attributs et méthodes des classes ne sont pas affichés ici.

[36] En fait, une interface de programmation, l'équivalent d'une classe abstraite.

Les classes Lemma et Synset sont affichées différemment pour marquer leur appartenance au lexique sémantique (présenté en partie IV) et non aux niveaux de représentation.

Nous allons à présent décrire les classes appartenant à chaque niveau de représentation. En partant du niveau texte, un Document est segmenté en plusieurs phrases (classe Sentence). Une phrase est elle-même associée à une ou plusieurs représentations (classe Analysis) de niveau morphologique, syntaxique ou sémantique, ce qui permet de gérer les ambiguïtés.

Au niveau morphologique de surface, une analyse est constituée *a minima* d'une liste de mots (Word) dont la forme de base et la partie du discours sont connues. La RMorphP se compose de syntagmes (classe Chunk) qui regroupent des mots.

Ce modèle de données linguistiques permet de stocker le résultat produit par un analyseur syntaxique en dépendances ou en constituants. La RSyntS est constituée de dépendances entre un mot gouverneur et un mot dépendant (classe Dependency) et/ou du nœud racine d'un arbre syntagmatique[37].

La classe DeepDependency stocke les dépendances syntaxiques profondes de la RSyntP. Leur regroupement forme des prédicats (classe Predicate) dont les arguments sont les rôles syntaxiques profonds (classe LogicalRole).

La RSém est constituée de trois catégories d'informations :

- Chaque prédicat de la RSyntP est associé à une ou (éventuellement) plusieurs acceptions (classe Frame) qui précisent les rôles thématiques (classe ThematicRole) des arguments du prédicat.
- Chaque mot ou expression multi-mots est associé à une liste de sens possibles (classe WordSense). Un système de score permet de conserver les ambiguïtés : un sens possible est donc un lien vers un lemme du lexique sémantique[38], pondéré par ce score.
- Enfin, le document contient une liste de chaînes de coréférences composées d'un ensemble de syntagmes qui font référence à la même entité (classe ReferringExpression) ; la classe ReferringLink permet de conserver la liste des antécédents possibles d'une anaphore.

D. Prise en compte du multilinguisme

1. Principes

Antelope a initialement été développée pour l'anglais, pour des raisons de disponibilité de ressources dans cette langue, sans se préoccuper de multilinguisme. Quand nous avons envisagé de traiter une deuxième langue –le français–, nous avons souhaité faire d'Antelope une plate-forme multilingue. Notre motivation était de permettre une mutualisation entre plusieurs langues du code de certains composants, ou au moins de certains algorithmes. L'intérêt est une amélioration de la capacité à maintenir du code ; une plate-forme multilingue évite de devoir maintenir plusieurs versions d'un même module, chacune d'entre elles étant adaptée aux spécificités d'une langue donnée.

[37] La classe SyntacticNode est une spécialisation de la classe Chunk, qui ajoute une relation récursive.
[38] Ou, dit autrement, vers un Synset de WordNet.

Nous estimons cet objectif globalement atteint. Par exemple, le code du composant d'extraction terminologique d'Antelope fait 400 lignes, et traite l'anglais et le français avec une version unique. A titre de comparaison, le programme Acabit (Daille, 1994), qui rend le même service, compte deux modules distincts (un pour l'anglais et l'autre pour le français) qui font chacun 4 000 lignes de code environ. Cette comparaison permet de souligner concrètement l'intérêt d'une plate-forme : disposer d'un environnement avec une bibliothèque de composants prêts à l'emploi, disponibles sur l'étagère, permet de développer rapidement des applications de TAL. Pour donner un second exemple, le composant de résolution d'anaphores d'Antelope opère sur les deux langues avec une version unique. Seulement une dizaine de lignes de code y sont dédiées à des spécificités de l'anglais et du français.

2. Déclaration d'une nouvelle langue dans Antelope

Antelope permet de définir les caractéristiques associées à chaque langue prise en compte. En pratique, à ce jour, seules des langues d'Europe de l'ouest ont été intégrées à la plate-forme. Nous nous inspirons en cela de (Chomsky, Lasnik, 1993) qui postule que la syntaxe d'une langue naturelle repose sur des principes universels, modulo des paramètres propres à chaque langue.

a) Parties du discours

Notre approche d'un traitement générique des langues passe par un certain nombre de choix. Nous commençons par définir dans Antelope un ensemble fermé de parties du discours, indépendantes de la langue[39] : nom, verbe, adjectif, adverbe, pronom, pronom possessif, déterminant, déterminant possessif, préposition, conjonction de coordination, conjonction de subordination, numérique, interjection, ponctuation et autre (mot étranger, formule mathématique…).

b) Traits morphosyntaxiques

Nous définissons ensuite un ensemble fermé de traits morphosyntaxiques : type de nom (commun, propre), personne, modalité verbale, temps verbal, degré de comparaison (comparatif, superlatif), définitude, type de nombre (cardinal, ordinal), genre et nombre. Notre source d'inspiration a été MAF (*Morpho-syntactic Annotation Framework*) décrit dans (Francopoulo *et al.*, 2008)[40].

c) Association de traits morphosyntaxiques aux parties du discours

Enfin, pour une langue donnée, nous associons des traits morphosyntaxiques à chaque partie du discours existant dans cette langue. Notre parti pris n'est pas de chercher *a priori* à définir un système universel applicable à toutes les langues, mais modestement (et c'est déjà suffisamment complexe) de proposer une solution opérationnelle capable de prendre en compte les langues traitées par la plate-forme (à l'heure actuelle, seules des langues européennes sont prévues).

Par exemple, en anglais, un déterminant a pour traits une définitude (permettant de différencier par exemple l'indéfini "a" et le défini "the") et un nombre (pour établir le contraste entre "this" au singulier et "these" au pluriel). Ce sera défini dans Antelope de la façon suivante :

---

[39] Toutes ces parties du discours ne se retrouvent pas nécessairement dans chaque langue ; par exemple, certaines langues n'ont pas de déterminant ou de pronom possessif.

[40] Une source d'inspiration similaire aurait pu être EAGLES (Calzolari *et al.*, 1996).

```
EnglishLanguage.SetPartOfSpeechFeatures(PartOfSpeech.Determiner,
{
	typeof(Number),
	typeof(Definiteness)
} );
```

En français, nous y ajoutons le genre (pour avoir par exemple « le » au masculin, « la » au féminin et « les » au masculin ou au féminin) :

```
FrenchLanguage.SetPartOfSpeechFeatures(PartOfSpeech.Determiner,
{
	typeof(Number),
	typeof(Definiteness),
	typeof(Gender)
} );
```

d) Adaptation des jeux d'étiquettes

Chaque analyseur existant a ses spécificités. Une intégration dans Antelope doit en tenir compte. Prenons l'exemple d'un étiqueteur morphosyntaxique : il opère sur une langue donnée (disons l'anglais), et annote les mots avec un jeu d'étiquettes spécifique à cette langue (Penn TreeBank par exemple). L'intégration de cet analyseur dans Antelope passe par la conversion de son jeu d'étiquettes spécifique vers les parties du discours génériques et les traits morphosyntaxiques prédéfinis d'Antelope.

e) Mots de classe fermée

Antelope permet de déclarer l'ensemble des mots de classe fermée. Par exemple, la déclaration du déterminant « les » en français est codée de la façon suivante :

```
FrenchLanguage.Declare(PartOfSpeech.Determiner, "les",
{
	features.Number = Number.Plural;
	features.Gender = Gender.Masculine | Gender.Feminine;
	features.Definiteness = Definiteness.Definite;
});
```

3. Niveaux de prise en charge d'une langue

La reconnaissance de la langue d'un texte est effectuée avec JLangDetect (Champeau, 2008) ; ce composant effectue un apprentissage de n-grammes sur les différentes langues du corpus EuroParl ; une opération similaire serait possible sur les Wikipédias, ce qui permettrait de couvrir un plus grand nombre de langues. Lors d'une analyse, chaque mot est associé à sa langue ; la forme de base est ensuite calculée à partir de la forme fléchie. Les lemmes du lexique sémantique ont aussi une information de langue, ce qui permet de faire le lien avec un mot analysé.

Le premier niveau de prise en charge d'une langue par Antelope consiste à savoir effectuer un certain nombre d'opérations linguistiques de base : segmentation en phrases et en mots, lexémisation, obtention de l'ensemble des mots de classe fermée d'une partie du discours donnée[41], conversion d'un jeu d'étiquettes spécifique à une langue (ou à un analyseur particulier) vers un jeu de parties du discours et de traits morphosyntaxiques génériques, indépendant de la langue. Ce

[41] Pour les mots qui ne sont pas des noms, verbes, adjectifs ou adverbes.

niveau suffit pour effectuer des regroupements de documents (*clustering* en anglais). Un deuxième niveau de prise en charge concerne des opérations plus complexes d'étiquetage morphosyntaxique, d'analyse syntaxique de surface, d'analyse syntaxique profonde et l'accès à un lexique sémantique de large couverture dans la langue concernée. Le troisième niveau concerne les opérations sémantiques : reconnaissance d'entités nommées, étiquetage de rôles sémantiques, résolution d'anaphores, désambiguïsation lexicale. Antelope couvre ces trois niveaux pour l'anglais et le français ; la prise en compte des principales langues européennes est actuellement en cours pour le deuxième niveau.

Lors de nos travaux, nous avons constaté la difficulté, voire l'impossibilité, d'obtenir des concepts universels et indépendants des langues. Les concepts (synsets) de WordNet sont construits par des locuteurs anglophones, et l'on peut émettre des réserves sur la pertinence de certains d'entre eux[42]. Certains concepts peuvent être raffinés dans une langue donnée : il existe ainsi plusieurs mots pour dire « riz » en japonais[43] ou « neige » en inuit[44] ; cela montre bien l'importance de ces concepts dans la langue et la culture concernées, qui peut déconcerter un français. Nonobstant ces exemples, il ne nous semble pas faux d'affirmer que toutes les langues occidentales partagent une écrasante majorité de concepts.

E. Capacité à préserver les ambiguïtés

La capacité à préserver les ambiguïtés (syntaxiques, lexicales, référentielles…) aussi longtemps que possible est un point important de la plate-forme. Par exemple, chaque phrase peut être associée à une ou plusieurs analyses ; l'une de ces analyses (indiquée par la relation *BestAnalysis*) est la meilleure, au sens d'un système de vote présenté en détail en VII.B, page 154.

F. Architecture technique

1. Environnement de développement .NET

Antelope est développée nativement en C#, un langage objet créé par Microsoft dans le cadre de son architecture .NET. L'ensemble constitué par C#, la bibliothèque de classes et la machine virtuelle[45] .NET, est proche de ce qui est proposé par Java. C# est un langage objet moderne, permettant le développement par classes et interfaces, une gestion des erreurs par exceptions, et une désallocation automatique de la mémoire grâce à un ramasse-miettes[46]. La version la plus récente introduit des éléments de programmation fonctionnelle (λ-expressions).

---

[42] Par exemple, le concept {PEOPLE OF COLOR} de WordNet nous semble *US centric*, voire *WASP centric*.

[43] Riz avant cuisson, riz cuit, riz du matin, riz du soir…

[44] Neige au sol, neige qui tombe, bourrasque de neige, amoncellement de neige…

[45] Une machine virtuelle est un interpréteur de code intermédiaire (appelé *Intermediate Language* en .NET, un pseudo assembleur de haut niveau). La machine virtuelle isole l'application en cours d'exécution des spécificités matérielles de l'ordinateur hôte ; l'intérêt de cette approche est d'éviter qu'une erreur applicative ne corrompe la mémoire de l'ordinateur.

[46] Le ramasse-miettes (*garbage collector* en anglais) est un mécanisme de gestion automatique de la mémoire ; ce mécanisme est transparent pour le développeur, qui se contente de créer des objets, sans les détruire explicitement ; le ramasse-miettes est responsable du recyclage de la mémoire occupée par les objets quand ils ne sont plus utilisés.

Antelope fonctionne sous Windows avec .NET, et aussi sous Linux avec MONO[47]. .NET semble moins utilisé que Java, C/C++, PERL ou Python dans la communauté du TAL. On peut toutefois noter que NooJ ou SharpNLP ont aussi fait ce choix.

2. Bonnes pratiques issues du génie logiciel

Le développement de grands systèmes d'informations, qui nécessite des milliers de jours d'analyse et de codage sous la pression de délais courts, a forcé à structurer des méthodes de travail. La notion de génie logiciel s'est progressivement imposée dans l'industrie informatique, visant à fournir des « bonnes pratiques » sous forme de méthodes de conception appelées *design patterns* en anglais (Gamma *et al.*, 1993). Elles sont aujourd'hui adoptées par les organisations ayant besoin de créer des applications de grande taille. Nous présentons ici les bonnes pratiques retenues pour la conception d'Antelope.

a) Modèle de programmation par interfaces

La programmation orientée objet est un paradigme qui a fait ses preuves. Les langages à objets récents (Java, C#...) ont introduit, en plus de la notion de classe[48], une notion explicite d'*interface*, un regroupement logique de propriétés et de méthodes. Une classe peut implémenter une ou plusieurs interfaces ; une interface peut être implémentée par plusieurs classes. Ce modèle de programmation systématise une séparation formelle entre interface et implémentation, favorisant un couplage faible entre composants, et donc une meilleure réutilisation.

Illustrons cette démarche sur un cas pratique. Par exemple, une opération d'étiquetage morphosyntaxique prend comme paramètre en entrée un texte (déjà découpé sous forme d'une liste de mots) et produit en sortie une liste d'étiquettes (chacune d'entre elles étant associée à un mot). On peut définir une interface `ITagger`[49] de la façon suivante :

```
List<Etiquette> Tag(List<string> mots);
```

Plusieurs implémentations sont évidemment possibles ; on peut imaginer coder l'étiquetage morphosyntaxique en utilisant des mécanismes tels que des modèles cachés de Markov (HMM), des séparateurs à vastes marges (SVM) ou encore des champs conditionnels aléatoires (CRF). On disposera alors d'autant de composants (`HmmTagger`, `SvmTagger`, `CrfTagger`…), respectant tous les spécifications de l'interface `ITagger`. Le reste de l'application manipulera ces composants d'une façon abstraite en tant que `ITagger`, sans avoir besoin de connaître leur implémentation particulière[50]. L'application utilisant ces composants peut alors très facilement substituer une implémentation à une autre ; cela permet de choisir, pour une tâche donnée, celui qui donne les meilleurs résultats en fonction de la nature des documents à analyser.

b) Définition de tests unitaires et de tests de non-régression

Un test unitaire est destiné à s'assurer du fonctionnement correct d'un composant d'un logiciel indépendamment du reste du programme, afin de vérifier qu'il répond aux spécifications prévues. Après une modification substantielle du code ou un changement de version d'un autre composant,

[47] MONO (http://www.mono-project.com) est un portage libre de .NET sur Linux et Mac OS.
[48] En C++, une interface peut se définir comme une classe abstraite virtuelle pure.
[49] Ici codée en langage C# ; par convention, le nom d'une interface commence par la lettre I en majuscule.
[50] Sauf évidemment lors de l'instanciation (il faut *quand même* préciser à un moment donné qui fait quoi), qui peut être effectuée avec un *design pattern* de type Factory ou un mécanisme d'injection de dépendances.

l'ensemble des tests unitaires peut être rejoué afin de rechercher d'éventuelles régressions du code (c'est-à-dire l'apparition d'erreurs nouvelles).

c) Conception technique permettant le passage à l'échelle

La capacité à « monter en charge » est un point essentiel pour traiter des corpus importants. L'environnement .NET dispose de nombreux protocoles de communication entre machines[51], qui permettent de distribuer facilement des traitements, et d'infrastructures de grilles de calcul massivement parallèle[52].

La course à la puissance des processeurs passe aujourd'hui davantage par la multiplication des « cœurs » sur un processeur que par l'augmentation de leur fréquence. Pour en tirer parti et ne pas sous-exploiter les architectures matérielles actuelles, il faut distribuer les calculs sur ces différents cœurs, en lançant des tâches multiples (*multithreading*) au sein d'un même processus. Ce type de développement est au centre des systèmes d'exploitation et des serveurs de bases de données. La conception de tels systèmes est connue depuis longtemps pour être complexe (Dijkstra, 1965). De plus, les outils qui aident à détecter un risque d'interblocage sont rares ; une erreur nouvelle risque toujours d'apparaître à l'exécution dans une configuration qui n'aura jamais été rencontrée lors des tests.

Les composants spécifiquement écrits pour Antelope ont été conçus pour s'exécuter dans un environnement multitâche. Les composants externes sont encapsulés avec un mécanisme qui garantit l'invocation séquentielle des méthodes. Nous avons pu, par exemple, effectuer une analyse syntaxique complète de la *Simple Wikipedia*[53] en trois jours. Pour cela, nous avons distribué Antelope sur cinq ordinateurs tournant avec un processeur à double cœur ; ils communiquaient (*via* des services Web) avec un serveur chargé de leur affecter des analyses de phrases et de consolider et stocker les résultats. Sur une seule machine mono-cœur, ce calcul aurait alors pris près d'un mois.

d) Sauvegarde et restauration du résultat d'une analyse

De même qu'un logiciel bureautique permet d'enregistrer un document sur disque puis de le rouvrir, Antelope peut sauvegarder le résultat de l'analyse d'un texte dans un fichier (typiquement dans un format XML), puis le recharger en mémoire. Ce mécanisme autorise le transport d'un résultat d'analyse entre machines, donc la répartition des traitements, ainsi que leur interruption et reprise.

e) Présence d'un mécanisme d'extensibilité par annotations

Les annotations sont un modèle très fréquemment utilisé, offrant un mécanisme d'enrichissement de l'information. Dans Antelope, ce modèle est utilisé pour le stockage (par exemple, dans des fichiers XML) et aussi, si nécessaire, pendant des opérations de calcul en mémoire. Il permet de greffer dynamiquement de nouvelles informations à des instances d'objets, sans imposer une modification du code de leur classe ou une spécialisation. Les principaux objets du modèle unifié d'Antelope (document, phrase, analyse syntaxique, mot, coréférence) ainsi que ceux du lexique sémantique (lemme, synset) disposent d'annotations[54]. Elles permettent de stocker :

---

[51] Web services, XML sur TCP, .NET *remoting* (flux binaire sur HTTP ou TCP).

[52] Par exemple www.alchemi.net ou www.digipede.net (Antelope ne les exploite pas encore).

[53] A cette époque, la *Simple Wikipedia* comportait 15 000 articles écrits en anglais simplifié.

[54] Une annotation est implémentée sous forme d'une structure de dictionnaire, c'est-à-dire une liste de paires (nom, valeur), avec la syntaxe objet.Annotations["nom"] = valeur.

- Des résultats intermédiaires pendant un calcul : par exemple, lors du calcul des anaphores, une annotation précise si un pronom est pléonastique ou non.
- Des données complémentaires optionnelles : un concept du lexique sémantique peut ainsi être relié à l'URL de l'article correspondant dans la Wikipédia.

3. Intégration de composants externes

L'équipe Proxem a développé ses propres analyseurs syntaxiques à partir de 2011, par apprentissage automatique : un étiquetage morphosyntaxique (en utilisant des CRF) et un analyseur robuste du français. Antelope intègre aussi plusieurs composants externes, codés en différents langages :

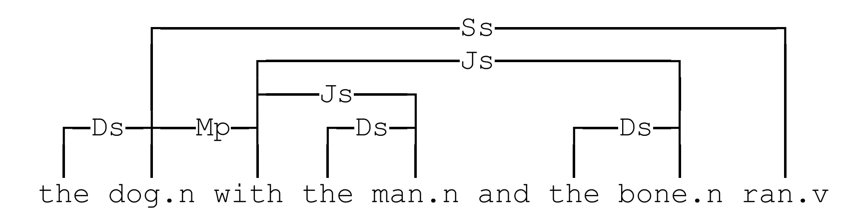
- L'étiqueteur morphosyntaxique SS Tagger[55] (composant C++).
- Un étiqueteur morphosyntaxique « à la Brill » (réimplémenté nativement en C#).
- Deux analyseurs syntaxiques robustes pour la langue anglaise, qui traitent avec succès des phrases complexes et tolèrent la présence de mots inconnus :
  - Le Stanford Parser (Manning, Klein, 2002) est un analyseur probabiliste écrit en Java, fourni avec plusieurs grammaires (allemande, chinoise, arabe). Il produit une forêt d'arbres de constituants, et sait les traduire en arbres de dépendances.
  - Le Link Grammar Parser (Sleator, Temperley, 1991), codé en C, repose sur des règles. Il produit un ou plusieurs arbres de dépendances (plus précisément, de liens typés reliant des paires de mots), puis les transforme en arbres de constituants.
- Un analyseur syntaxique du français : le TagParser[56] (Francopoulo, 2008), codé en Java, qui produit comme analyse un arbre de dépendances entre *chunks*.

Antelope utilisant systématiquement le modèle de programmation par interfaces, ces composants externes sont encapsulés par une interface `ITagger` ou `IParser`, ce qui permet de les rendre interchangeables.

a) Link Grammar Parser

Partant d'une phrase, cet analyseur en détermine la structure syntaxique, qui consiste en un ensemble de liens typés reliant des paires de mots. La grammaire de dépendances utilisée pour l'anglais distingue 107 types de liens. L'analyse peut donner une forêt d'arbres, chaque arbre étant pondéré par un « coût syntaxique ». Par exemple, la phrase "*the dog with the man and the bone ran*" donne deux analyses, correspondant à la distribution possible autour de la conjonction de coordination and :

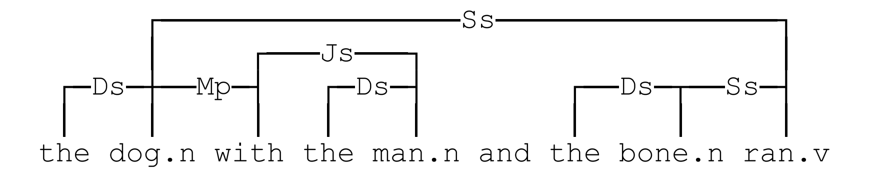
The dog with (the man and the bone) ran



[55] Un étiqueteur morphosyntaxique rapide qui utilise une extension des chaînes de Markov à entropie maximale. Voir (Tsuruoka, Tsujii, 2005).

[56] À titre indicatif, l'intégration de cette ressource a nécessité moins d'une semaine de travail.

(The dog with the man) and (the bone) ran



Dans les arbres de dépendances produits par le Link Grammar Parser, on remarque que :

- Les nœuds sont les mots de la phrase ; certains d’entre eux ont un suffixe qui indique la partie du discours (nom, verbe, adjectif, adverbe, préposition, etc.). Par exemple, “*ran.v*” est reconnu en tant que forme de verbe (suffixe .v), et “*dog.n*” en tant que forme de nom (suffixe .n).
- Des arcs étiquetés relient les nœuds du graphe ; chaque étiquette précise un rôle grammatical (sujet, déterminant, etc.). Par exemple, entre “*dog.n*” et “*ran.v*”, le libellé du lien est « Ss » où :
    - La première lettre (S_ majuscule) désigne une fonction sujet (***S**ubject*).
    - La seconde lettre (_s minuscule) précise qu’il est singulier (***s**ingular*).

(1) Robustesse et rapidité de l’analyseur

L’analyseur offre de bonnes performances pour un analyseur en profondeur, ainsi qu’une excellente robustesse. Il traite avec succès des phrases complexes, et est tolérant à la présence de mots inconnus. Quand il en rencontre, il essaie de déterminer leur partie du discours en fonction du contexte. Par exemple, dans la phrase suivante, `machining` n’est pas présent dans le lexique de référence (comme l’indique l’annotation [!]), mais est supposé être le gérondif d’un verbe (extension .g).

```
    +---Ss---+----Pg*b----+----MVp---+-Js-+
    |        |            |          |    |
George.n was.v machining[!].g in Najaf
```

(2) Stockage externe de la grammaire

La LinkGrammar s’appuie sur une grammaire lexicalisée qui couvre d’une façon relativement complète la langue anglaise courante. Chaque élément du lexique est associé à un ensemble de structures élémentaires qui sont des configurations de la structure de dépendance décrivant les liens possibles de cette unité lexicale. Le formalisme est équivalent à une CFG et l’algorithme d’analyse est en $O(n^3)$.

La grammaire est stockée indépendamment du code, dans des fichiers textes qui contiennent également le lexique de référence (approximativement 60 000 mots). Ce stockage externe permet de modifier ou d’enrichir la grammaire ou le lexique. En revanche, le formalisme de stockage est propriétaire et plutôt complexe à maîtriser.

Nous avons importé la ressource LinkGrammar-WN, qui enrichit le lexique du Link Grammar Parser de 14 000 nouveaux noms provenant de WordNet. Elle a été créée en se basant sur la démarche de fusion de lexiques présentée dans (Szolovits, 2003).

b) Mécanismes techniques d'intégration

L'intégration des composants externes se fait de différentes façons. Si le code source du composant est disponible et écrit dans un langage pour lequel il existe un compilateur pour .NET[57], une recompilation suffit.

Si le composant est écrit en Java, nous utilisons le logiciel IKVM, qui est capable de traduire un fichier binaire Java `.class` ou `.jar` dans le *bytecode* équivalent de .NET. C'est le mécanisme que nous avons mis en œuvre pour intégrer les analyseurs d'Antelope à l'architecture UIMA (voir page 36).

Pour du code en C ou C++, nous encapsulons le composant à travers un mécanisme standard de bibliothèque dynamique (DLL) ; l'interface de programmation du composant est présentée en utilisant un *design pattern* classique (de type Façade), qui regroupe l'ensemble des services du composant dans une classe unique. L'expérience nous a toutefois montré qu'il est dangereux de mélanger du code .NET ou Java avec du code C ou C++ ; en effet, le premier s'exécute en bénéficiant de la protection offerte par une machine virtuelle : même en cas d'erreur d'exécution dans le programme, les mécanismes d'exceptions permettent une reprise sur erreur ; en revanche, une erreur dans du code C/C++ peut résulter en une corruption de la mémoire et une erreur fatale au programme. C'est donc difficile à accepter dans un contexte industriel, avec des contraintes de production en 24/7 [58]. Concrètement, nous avons constaté ce problème avec un correcteur orthographique ; une solution est alors d'effectuer un portage du code C/C++ vers du code natif C#.

c) Adaptation à un format commun

Il est relativement aisé d'intégrer un nouvel étiqueteur morphosyntaxique pour l'anglais, dans la mesure où le jeu d'étiquette du *Penn TreeBank* est un standard de fait, généralement bien suivi (à quelques détails près parfois, comme des étiquettes particulières pour les verbes *to be* et *to have*).

L'effort à fournir est plus important pour les analyseurs syntaxiques. En effet, même s'ils produisent des structures de même nature (arbres de constituants et arbres de dépendances) avec des étiquettes de constituants normalisées (NP, VP, PP…), les étiquettes et l'organisation des dépendances diffèrent radicalement entre les différents analyseurs que nous avons utilisés.

La figure 5 permet de le constater, en comparant la RSyntS produite par le Link Grammar Parser (à gauche) avec celle du Stanford Parser (à droite), lors de l'analyse syntaxique d'une même phrase (l'arbre de constituants est au-dessus des mots, l'arbre de dépendances en-dessous). Nous verrons au chapitre VII.A que lors du passage en RSyntP nous obtenons un format commun de dépendances syntaxiques profondes.

---

[57] VB, Pascal, Python, Eiffel, COBOL, PHP, etc. étendus pour offrir des fonctionnalités identiques.

[58] Abréviation pour « 24 heures sur 24, 7 jours sur 7 », qui signifie que le service est disponible en permanence.

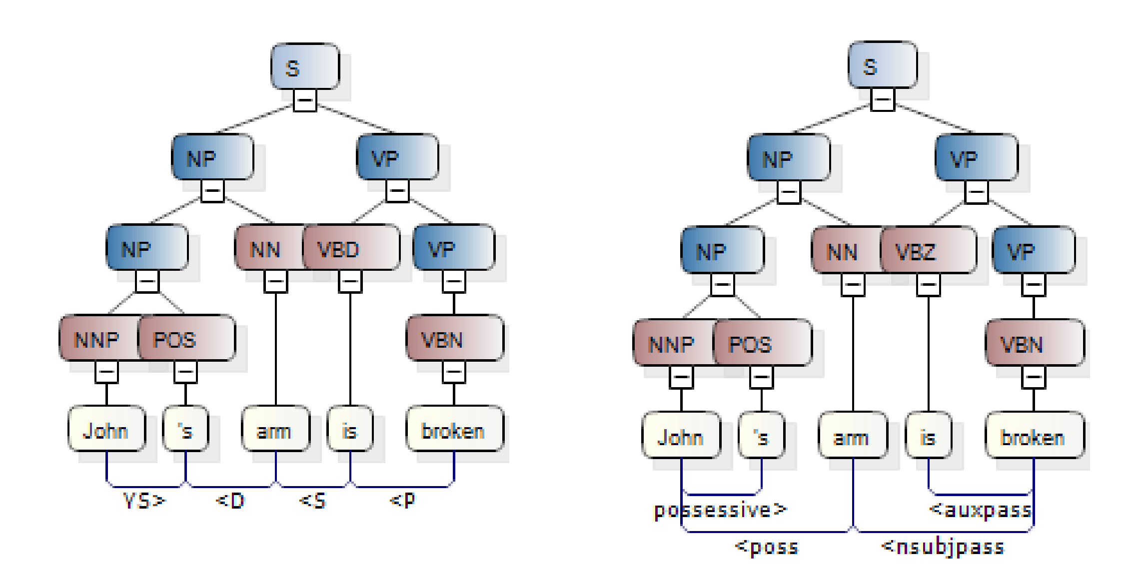


Figure 5 : Comparaison des sorties du Link Grammar et du Stanford Parser

G. Positionnement par rapport à d'autres plates-formes

Cette section présente brièvement des architectures et plates-formes de référence de traitement du langage, en positionnant Antelope par rapport à celles-ci. Au vu des caractéristiques de ces plates-formes, il nous semble que la principale originalité d'Antelope réside dans l'intégration d'un lexique sémantique à large couverture, dans son modèle en niveaux de représentations clairement définis et dans la présence d'une ISS.

1. GATE

GATE (*General Architecture for Text Engineering*) (Cunningham *et al.*, 1996) est une infrastructure permettant le développement et le déploiement de composants pour le traitement de la langue naturelle. Développée depuis 1995 à l'Université de Sheffield, elle est largement utilisée sur des tâches de fouille de textes et d'extraction d'information. GATE propose une architecture, un *framework* en Java (incluant de nombreux modules) et un environnement de développement intégré.

GATE intègre en standard plusieurs composants linguistiques qui effectuent des tâches de segmentation, d'étiquetage morphosyntaxique, de détection de coréférences, d'identification d'entités nommées, d'extraction d'information et d'analyse syntaxique. Ces différentes tâches produisent des annotations sur les documents.

2. OpenNLP

Projet incubé à la Fondation Apache, OpenNLP est une boîte à outil *open source* pour le TAL, codée en Java ; OpenNLP contient des modules de segmentation, étiquetage morphosyntaxique, *chunking*, analyse syntaxique en constituants, détection d'entités nommées et extraction des coréférences ; ces différents modules se basent sur la librairie Java d'apprentissage *OpenNLP.Maxent*, qui utilise un modèle de maximisation d'entropie (Ratnaparkhi, 1996). La conception d'ensemble d'OpenNLP et sa

couverture nous paraissent proches de celles d'Antelope. Nous disposons toutefois d'une ISS et d'analyseurs syntaxiques en dépendances absents d'OpenNLP.

3. LinguaStream

LinguaStream (Bilhaut, Widlöcher, 2006) est une plate-forme générique pour le TAL, développée en Java au GREYC depuis 2001. Son environnement de développement intégré permet de créer visuellement des chaînes de traitement linguistique complexes, en assemblant des modules de différents niveaux. Chaque maillon de la chaîne peut annoter le document. LinguaStream facilite la réalisation d'expériences sur corpus, en ne requérant que peu de compétences informatiques.

Le public visé par les deux plates-formes n'est pas exactement identique. Pour Antelope, il s'agit essentiellement de développeurs informaticiens ; la cible de LinguaStream est peut-être davantage constituée de linguistes informaticiens désireux de réaliser facilement des expérimentations sur corpus.

4. LingPipe

LingPipe est une bibliothèque commerciale Java qui permet de traiter des corpus en langue anglaise ou chinoise. LingPipe permet de réaliser les traitements linguistiques suivants : conversion d'un texte html en xhtml, segmentation d'un texte en phrases avec prise en compte des acronymes, étiquetage morphosyntaxique, reconnaissance d'entités nommées (lieux, personnes…), résolution d'anaphores pronominales et de coréférences.

LingPipe se base sur des exemples d'apprentissage pour construire certains de ses modèles. LingPipe est notamment utilisée en bioinformatique (Carpenter, 2007).

5. UNL

UNL (Hiroci *et al.*, 1999 ; Sérasset, Boitet, 2000) n'est pas stricto sensu une plate-forme, mais plutôt une langue artificielle pouvant être utilisée comme formalisme de représentation des connaissances ou comme langage pivot interlingue en traduction automatique ; néanmoins, une plate-forme de développement a été bâtie autour. UNL a été conçu pour la compréhension comme pour la génération de texte. En pratique, la stratégie de développement porte actuellement plus sur la génération d'un énoncé en langage UNL vers une langue naturelle ; la compréhension de texte est aujourd'hui envisagée avec une approche semi-automatique, avec une validation humaine interactive. L'objectif principal d'UNL est donc de favoriser la traduction d'un énoncé en plusieurs langues[59].

UNL représente un texte, phrase par phrase, comme un hypergraphe composé d'un ensemble de liens étiquetés dirigés (les relations) entre les nœuds ou hypernœuds (« mots universels » : *Universal Words* ou UW), qui représentent les concepts. Les UW peuvent aussi être annotés avec des attributs contenant des informations de contexte.

[59] UNL est un programme issu de l'Université des Nations Unies, une agence de l'ONU créée en 1973, qui a notamment pour objectif d'établir des relations entre l'ONU et la communauté universitaire. L'intérêt d'une automatisation des traductions est évident pour l'ONU. UNL revendique une ambition de couverture d'un grand nombre de langues, mais il semble que les travaux soient surtout actifs sur l'anglais et le japonais.

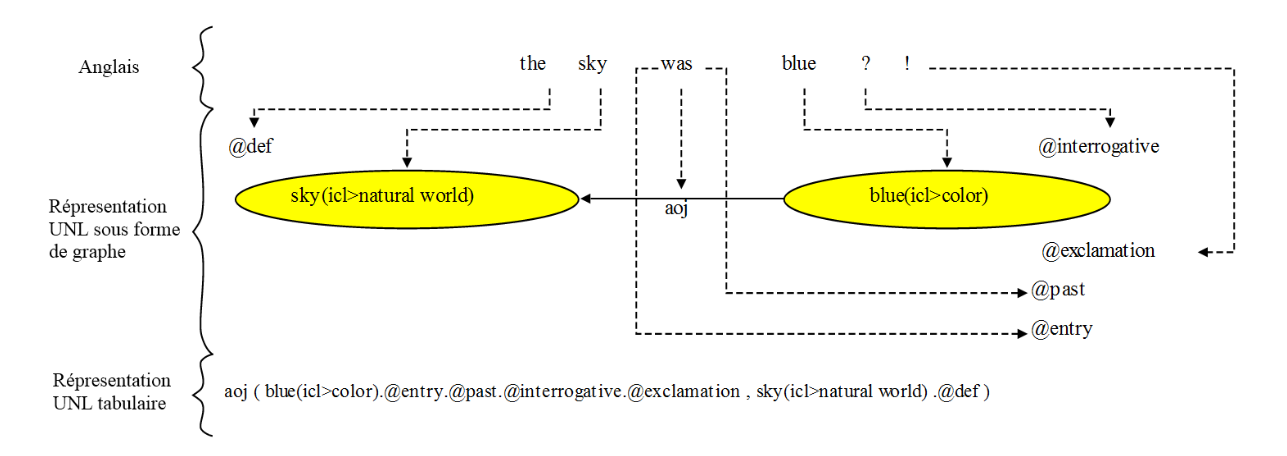


Figure 6 : Représentation UNL de la phrase anglaise « *the sky was blue?!* »

Dans l'exemple de la figure 6 ci-dessus, `sky(icl>natural world)` et `blue(icl>color)` représentent des concepts différents et sont des UW ; `aoj` (*attribute of an object*) est une relation binaire sémantique dirigée reliant les deux UW ; `@def`, `@interrogative`, `@past`, `@exclamation` et `@entry` sont des attributs modifiant les UW.

Les UW représentent des **concepts universels**, exprimés par des mots en anglais ou dans toute autre langue naturelle, lisibles par un humain. Ils se composent d'une tête (la racine de l'UW) et d'une liste de contraintes (le suffixe entre parenthèses) permettant de lever l'ambiguïté de la tête. L'ensemble des UW est organisé sous forme d'une ontologie (*UW System*), où les concepts du haut sont utilisés pour lever l'ambiguïté de leurs hyponymes grâce aux relations `icl` (est une sorte de), `iof` (est une instance de) et `equ` (est égal à).

Les relations représentent les **liens sémantiques** entre mots dans toutes les langues. Elles peuvent être ontologiques (comme `icl` et `iof`), logiques (comme `and` et `or`) ou actancielles (comme `agt` = *Agent*, `ins` = *Instrument*, `tim` = *Temps*, `plc` = *Lieu*…). La spécification UNL compte actuellement 46 relations, qui définissent sa syntaxe.

Les **attributs** donnent des informations qui ne sont pas portées par les UW et les relations, par exemple sur le temps (`@past`, `@future`…), la détermination (`@def`, `@indef`…), la modalité (`@can`, `@must`…), le sujet de la discussion (`@topic`, `@focus`…).

H. Compatibilité avec l'architecture UIMA

1. Principes d'UIMA

UIMA (Ferrucci D., Lally A., 2004) est l'un des efforts les plus aboutis pour rendre interopérables des composants de TAL. Cette architecture est née d'un besoin interne d'IBM Research, qui comptait plus de 200 personnes travaillant sur des sujets très variés de TAL : recherche d'information, détection d'entités nommées, classification de documents, traduction automatique, questions-réponses… D'une part, cette diversité a poussé différentes équipes à réfléchir au meilleur moyen de partager leurs résultats. D'autre part, la possibilité de réutiliser et de combiner des résultats d'analyse grâce à une architecture commune et un cadre logiciel robuste est de nature à permettre d'intégrer plus rapidement les résultats des équipes de R&D dans les produits logiciels d'IBM. Ces besoins ont conduit à l'élaboration d'UIMA (*Unstructured Information Management Architecture*, architecture de

traitement des informations non structurées), qui offre des capacités de recherche d'information et une plate-forme de développement, de composition et de déploiement de moteurs d'analyse.

UIMA propose un cadre technique de référence, centré sur l'annotation de documents. Son objectif est de décrire les étapes de traitement d'un document de type texte, image ou vidéo afin d'en extraire de façon automatique des informations structurées. En revanche, UIMA ne décrit ni comment ces informations doivent être extraites, ni la façon de s'en servir. Cette architecture reste donc à un niveau très générique sur la notion d'annotation et ne propose pas de modèle de référence destiné à stocker les résultats des différents types d'analyses. Plusieurs composants annoteront donc successivement (ou en parallèle) des textes, mais ils ne pourront pas facilement partager les intermédiaires de calcul déjà effectués s'il n'y a pas de définition préalable d'un « CAS » (Common Analysis System) standard.

IBM a créé une implémentation de référence *open source* d'UIMA en C++ et en Java, avant de la transférer à la fondation Apache. L'ambition d'UIMA est de s'imposer en tant que standard industriel et norme ; UIMA a d'ailleurs été approuvée par l'OASIS en 2009.

2. Intégration d'Antelope à l'architecture UIMA

Antelope a des objectifs moins universels qu'UIMA et ne traite que l'analyse de documents textuels, orientée vers l'extraction de connaissances. Le modèle unifié d'Antelope est conçu sur mesure pour assurer cette tâche ; il ne s'agit donc pas d'un métamodèle, comme c'est le cas dans UIMA. On peut souligner une similarité d'architecture entre UIMA et Antelope : dans les deux cas, la conception est orientée composants et s'appuie sur un modèle de programmation par interfaces, avec des structures extensibles.

(Chaumartin *et al.*, 2009) présente en détail les modalités d'intégration d'Antelope à l'architecture UIMA. Une difficulté technique à résoudre était l'intégration des composants d'Antelope (conçus pour .NET) dans l'architecture UIMA, dont seules des implémentations C++ et Java existent. Réécrire l'ensemble des composants d'Antelope dans ces langages était inenvisageable. Nous avons donc cherché comment créer un annotateur UIMA, fonctionnant en .NET et non en Java, capable d'être appelé depuis n'importe quel processus client UIMA. Nous avons d'abord essayé d'exposer un service Web, mais cette approche n'a pas abouti[60]. Nous avons ensuite exploré une solution de plus bas niveau en utilisant un protocole d'appel entre *sockets* pour communiquer entre la machine virtuelle .NET et la machine virtuelle Java (JVM). Lors de ces essais, nous avons identifié un protocole standard d'UIMA, nommé Vinci, utilisant uniquement les *sockets* et des bibliothèques Java standards ; ce protocole était donc relativement facile à transposer en .NET. Nous avons utilisé IKVM pour convertir les bibliothèques UIMA (fichiers `.jar`) dans leur équivalent en .NET, ce qui nous permet au final d'invoquer les analyseurs d'Antelope. La figure 7 illustre cette architecture technique.

L'objectif de l'application SCRIBO (voir page 132) est l'extraction d'information à partir de dépêches de l'AFP. SCRIBO utilise plusieurs annotateurs en architecture UIMA, dont ceux d'Antelope.

[60] Le protocole SOAP est censé permettre une telle interopérabilité. Néanmoins, des problèmes d'incompatibilité entre les services Web exposés par UIMA et leur mise en œuvre en .NET sont apparus. En effet, les services Web d'UIMA exposent certaines classes utilisant les bibliothèques Axis, qui ne peuvent pas être facilement traduites dans leur équivalent .NET.

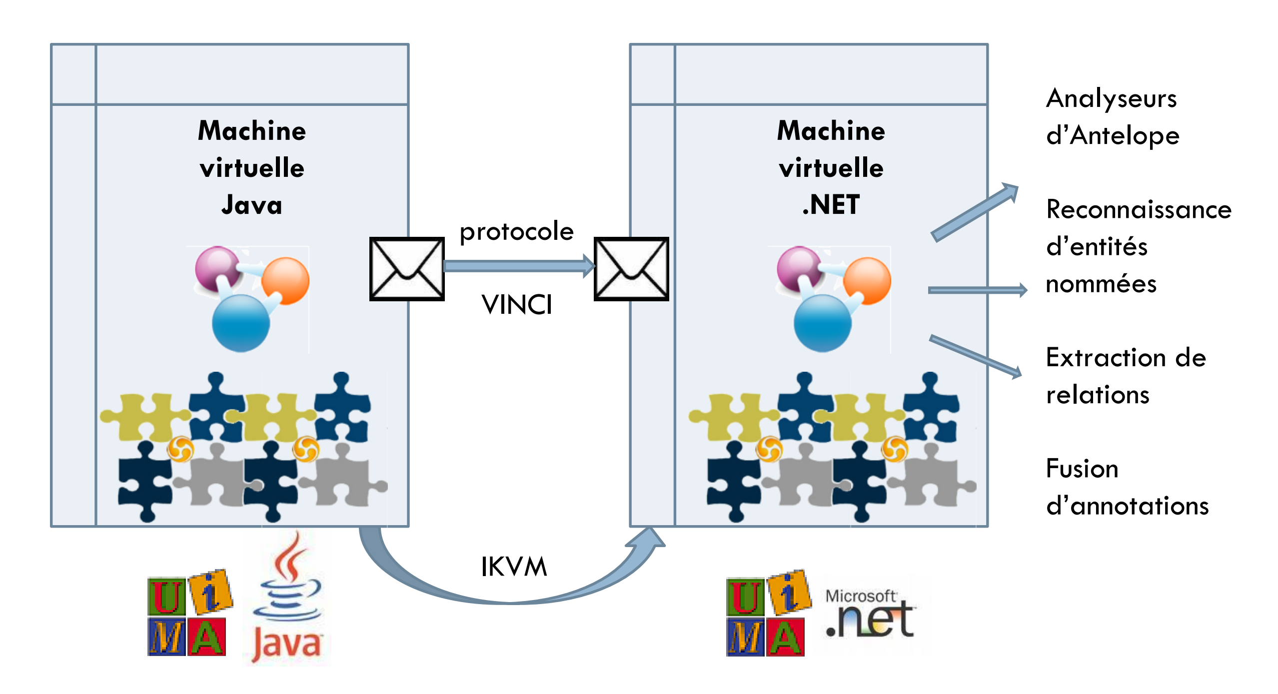


Figure 7 : Architecture technique permettant l'appel d'analyseurs écrits en .NET à partir d'UIMA

I. Composants de traitement jusqu'à l'analyse syntaxique

1. Nettoyage de documents (« *templating* »)

Le nettoyage de pages Web (*templating* ou *scrapping* en anglais) a pour objectif de diminuer le bruit dans les documents Web après leur collecte et avant leur analyse. En effet, les pages Web contiennent souvent des données qui « parasitent » l'élément principal de la page, comme les menus, les liens vers d'autres pages liées, ou encore des liens commerciaux. Le nettoyage permet de sauvegarder une copie locale « allégée » des pages Web (avec un volume stocké moins important) ; l'analyse de cette copie est plus pertinente. Nous avons exploré plusieurs pistes pour effectuer un tel nettoyage. Cette tâche n'étant toutefois qu'un préalable à celles du TAL, nous ne les décrirons pas plus en détail ici.

2. Segmentation et traitement de l'enrichissement typographique

Antelope traite des documents au format texte brut ou au format HTML. Dans ce dernier cas, un traitement préalable sépare le texte de son enrichissement typographique ; l'analyse des balises HTML permet un découpage du document en paragraphes. Ces paragraphes sont ensuite découpés en phrases en utilisant un ensemble de règles. L'information permettant de relier les mots, phrases et paragraphes aux balises est par la suite toujours disponible, ce qui permet :

- De déterminer si un paragraphe est un titre ou un élément d'une énumération[61].
- D'identifier les références quand le document contient des liens hypertexte[62].

[61] Auquel cas, il nécessite peut-être un traitement particulier : « décapitalisation » des initiales pour un titre, analyse syntaxique de phrase averbale pour un élément d'énumération.

3. Étiquetage morphosyntaxique, chunking ou analyse syntaxique

En fonction de ses besoins de vitesse ou de précision, l'utilisateur peut choisir entre un étiquetage morphosyntaxique, un *chunking* ou une analyse syntaxique. Antelope utilise pour cela les composants externes présentés en section F.3. Dans les trois cas, le modèle unifié décrit en section III.C (page 23) est alimenté, mais seules les représentations concernées (RMorphS, RMorphP ou RSyntS) sont renseignées.

4. Identification des expressions multi-mots

Antelope utilise le lexique sémantique pour identifier les expressions multi-mots. La forme de base de chaque mot est d'abord calculée par le module morphologique du lexique ; le composant teste la présence de *n*-uplets dans le lexique (*n* variant de cinq jusqu'à deux). Des règles supplémentaires permettent d'identifier aussi des expressions multi-mots non contigües, comme dans « *Pierre and Marie Curie* », « *Alabama and Mississippi Rivers* » ou « *the canon and civil law* ».

Lors d'une analyse syntaxique (par opposition à un simple étiquetage morphosyntaxique), une contrainte supplémentaire de rattachement est appliquée. Les mots ne sont regroupés que s'ils appartiennent à un même sous-arbre. Comme le montre la figure 8, l'expression « *Battle of Gettysburg* » est reconnue dans l'analyse syntaxique de gauche, mais pas dans celle de droite (absence de tête commune). Cela contribue à lever certaines ambiguïtés syntaxique (ce point est décrit page 162 en section VII.B.6.c).

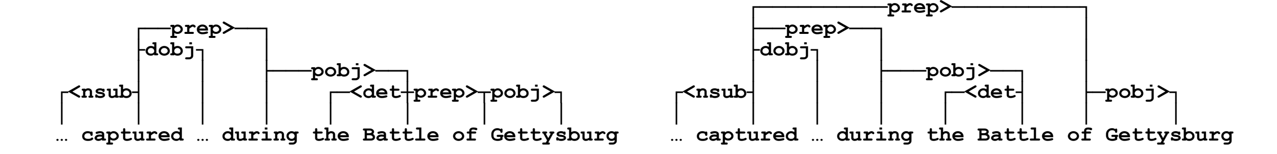


Figure 8 : Identification de l'expression multi-mots « *Battle of Gettysburg* »

J. Evolutions de la plate-forme

Antelope a été initialement conçue pour s'exécuter sur un poste de travail ou sur un serveur unique. Avec l'augmentation du nombre de projets et de la taille des corpus à traiter (plusieurs millions de documents), de nouveaux besoins sont apparus : pouvoir traiter un corpus volumineux sur une ferme de serveurs ; traiter plusieurs corpus simultanément ; exploiter la puissance et la souplesse du *cloud computing*[63]…

---

[62] Les liens hypertexte donnent un indice important de désambiguïsation lexicale (pour les entités nommées dans les articles d'encyclopédie, par exemple).

[63] L'*informatique dans le nuage* consiste à déporter sur des serveurs distants des stockages et des traitements informatiques traditionnellement localisés sur des serveurs locaux ou sur le poste de l'utilisateur. L'accès à des ressources virtualisées et mutualisées s'effectue à la demande, via Internet et en libre-service. L'intérêt pratique est par exemple de louer plusieurs dizaines de machines pour une durée limitée, sans avoir à les acheter, afin d'effectuer ponctuellement des calculs lourds. Le choix de Proxem s'est porté en l'occurrence sur Azure, la plate-forme publique de *cloud computing* de Microsoft.

En 2011, l'équipe Proxem a fait évoluer Antelope pour permettre à ses composants de s'exécuter sur des environnements différents, allant du poste de développeur au *cloud computing* en passant par la ferme de serveurs. Le fait d'utiliser un modèle de programmation par interfaces a grandement facilité cette transition. Antelope s'appuye désormais sur les principes d'inversion de contrôle[64] et d'injection de dépendances[65] (Fowler, 2004), qui permettent d'atteindre les différents objectifs précités tout en maintenant la complexité d'ensemble à un niveau raisonnable.

Cette évolution a nécessité en premier de séparer l'implémentation de l'environnement d'exécution (appelé conteneur) de celle des composants. L'équipe Proxem a conçu un conteneur spécialisé pour chaque environnement cible, et fait évoluer les composants d'analyse pour les rendre compatibles avec les contraintes de chaque environnement. La décision d'assemblage des composants pour former une configuration applicative se fait désormais uniquement au niveau du conteneur, d'une façon explicite. Les dépendances entre composants ne sont donc plus exprimées de façon statique dans le code, mais déterminées dynamiquement à l'exécution, ce qui permet de les modifier sans recompilation.

---

[64] L'inversion de contrôle (*Inversion of Control* ou plus simplement *IoC* en anglais) est un patron d'architecture commun à plusieurs boîtes à outils logicielles. Son principe est de faire en sorte que le flot d'exécution du code n'est plus sous le contrôle direct de l'application elle-même mais de la boîte à outil sous-jacente.

[65] L'injection de dépendances (*Dependency Injection* ou *DI* en anglais) est un mécanisme qui permet d'implémenter le principe de l'inversion de contrôle. Il consiste à créer dynamiquement (injecter) les dépendances entre les différentes classes en s'appuyant sur une description externe (typiquement stockée dans un fichier de configuration).

Partie IV. Lexique sémantique multilingue à large couverture

A. Introduction

Le lexique joue un rôle central dans une ISS. En effet, un lexique riche dispose d'informations lexicales (langues, expressions multi-mots, différents sens d'un mot, domaines…), syntaxiques (distributions statistiques d'usage, cadres de sous-catégorisation…) et pragmatiques (connaissance du monde, axiomatique pour effectuer des raisonnements logiques…). La précision des phénomènes décrits varie énormément d'un lexique à l'autre : *a minima*, il peut se constituer d'une liste de centaines de milliers de formes de surface existant dans une langue donnée, sans autre information ; à l'autre bout du spectre, certains lexiques sémantiques ne décrivent que quelques centaines d'entrées lexicales, mais d'une façon extrêmement précise.

Notre objectif est de traiter de « vrais » textes (articles de presse, offres d'emploi, opinions exprimées par des consommateurs…) en français et en anglais. Nous souhaitons donc disposer d'un lexique doté d'une couverture aussi large que possible. Pour cela, nous avons créé un lexique sémantique multilingue à large couverture en intégrant plusieurs ressources hétérogènes.

La constitution d'un lexique regroupant des informations aussi variées n'est pas chose aisée. Les lexiques électroniques sont développés avec différents formalismes, en intension ou en extension, dans différents formats. XML s'impose progressivement comme format d'échange ; les formats émergents du Web sémantique (RDF, RDFS, SKOS, OWL…) jouent un rôle d'importance grandissante pour représenter les lexiques, thésaurus et autres ontologies ; c'est pourquoi nous leur consacrons une annexe, page 187.

Rendre interopérables ces données lexicales nécessite généralement un travail d'ingénierie important, et souvent aussi un travail conceptuel d'adaptation pour établir une correspondance entre les entrées de deux lexiques. Le cœur de notre lexique sémantique est WordNet, développé pour l'anglais à l'Université de Princeton (Miller, 1995 ; Fellbaum, 1998). Dans le chapitre B, nous présentons en détail WordNet, puis l'écosystème des autres ressources qui gravitent directement autour. Nous les avons intégrées au sein d'une base unique pour en faciliter l'utilisation dans la plate-forme Antelope.

Le chapitre C décrit plusieurs expériences qui nous ont permis d'étendre ces données, d'une façon endogène ou exogène (notamment à partir d'articles encyclopédiques). Nous avons enrichi les entrées lexicales existantes ; nous en avons aussi ajouté de nouvelles à partir d'autres sources ; nous avons enfin créé de nouveaux types de relations, notamment de polysémie régulière.

Enfin, dans le chapitre D, nous présentons succinctement des ressources que nous prévoyons d'intégrer dans le futur en indiquant quel en serait l'intérêt pour les traitements de la plate-forme.

1. Dictionnaires, lexiques, taxonomies et ontologies

a) *Diversité des informations représentées*

Les données lexicales prennent des formes très variées, allant de la simple liste de mots au lexique sémantique en passant par le thésaurus. Elles concernent traditionnellement des aspects phonologiques, lexicaux, définitoires, morphologiques, pragmatiques, encyclopédiques ou sémantiques ; d'autres types d'informations sont apparues récemment en TAL, par exemple les émotions ou sentiments associés aux sens des mots.

En plus des ressources linguistiques décrivant des mots et les relations qu'ils entretiennent, les encyclopédies électroniques proposent des connaissances générales sur le monde, généralement sans formalisme structuré. Les taxonomies organisent ces connaissances en arborescence, le plus souvent au sein d'un domaine restreint ; dans ces arbres, les nœuds proches de la racine représentent les concepts les plus généraux, et les feuilles les concepts les plus spécifiques.

Les ontologies enrichissent les taxonomies avec une axiomatique, c'est-à-dire un ensemble de relations et de formules logiques qui décrivent les contraintes existant entre les concepts. L'objectif premier d'une ontologie est donc de modéliser un ensemble de connaissances dans un domaine donné. Son second objectif est de permettre d'effectuer des raisonnements sur les objets du domaine concerné, pour vérifier notamment que les contraintes sont bien respectées. La figure 9 montre, à titre indicatif, un fragment de l'ontologie SUMO.

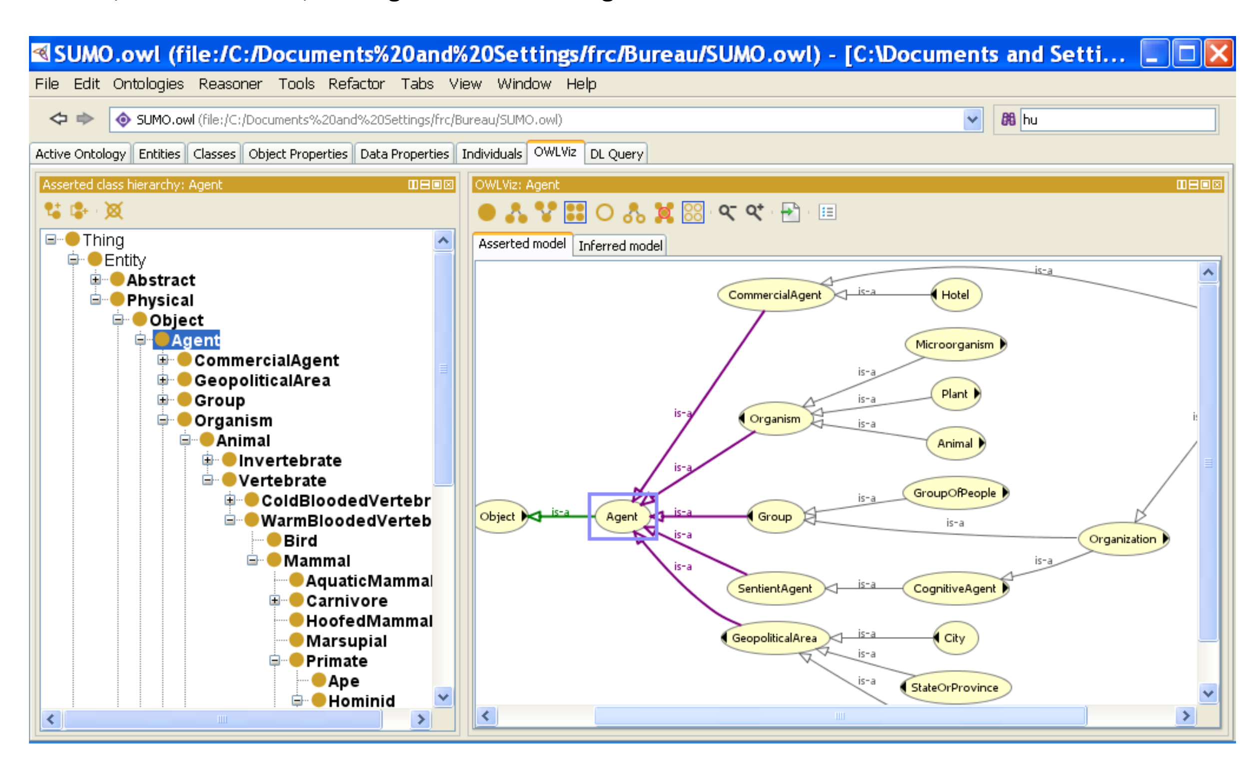


Figure 9 : Une partie de l'ontologie SUMO (affichée dans l'éditeur d'ontologie Protégé)

b) *Variété des formats et des encodages*

A la diversité des informations représentées s'ajoute celle des formats : ces ressources sont disponibles sous différentes formes, allant du simple fichier texte, avec une ligne par élément décrit, à des représentations structurées en XML. L'encodage même des caractères a longtemps été source de disparités, avec des cohabitations difficiles (ASCII, ANSI, ISO 8859-1…).

La généralisation d'Unicode représente une solution opérationnelle à ce problème ; Unicode ambitionne d'être un sur-ensemble de tous les autres encodages avec un répertoire complet contenant (mi-2012) autour de 110 000 caractères. Précisons que les chaînes de caractères sont représentées en Unicode dans Antelope, en mémoire comme dans les données persistantes. Cela permettra dans le futur la prise en compte de langues utilisant des alphabets non-européens.

2. Survol de ressources à large couverture fréquemment utilisées en TAL

Quelques ressources sont progressivement devenues des standards de fait en TAL. Elles ont pour caractéristiques communes d'être libres de droit, de proposer une large couverture d'une (ou plusieurs) langue(s), et d'être suffisamment structurées pour être facilement utilisées en TAL.

Ces différentes ressources sont décrites dans cette partie, ainsi que leur utilisation dans le cadre de nos travaux. Nous avons essentiellement mis en œuvre WordNet (et des extensions à WordNet), des Wikipédias en anglais et en français et l'ontologie SUMO.

Le tableau 2 montre la progression des citations de ces ressources dans CiteSeer entre juillet 2010 et juillet 2011. On voit que WordNet est probablement aujourd'hui la ressource la plus utilisé en TAL[66].

	WordNet	Wikipédia	CYC	Thésaurus Roget	Ontologie SUMO	Ontologie DOLCE
Juillet 2010	6 492	3 538	2 650	450	353	323
Juillet 2011	7 367	5 602	2 947	460	414	363
Progression	+13 %	+58 %	+11 %	+2 %	+17 %	+12 %

Tableau 2 : Evolution des citations dans CiteSeer de différentes ressources lexicales

a) Thésaurus de Roget

L'une des plus anciennes développées pour l'anglais est probablement le thésaurus de Roget. Dans sa première édition (1852), il comptait 15 000 mots anglais, organisés en six classes principales. La version utilisée en TAL est celle distribuée depuis 1996 dans le cadre du projet Moby ; elle enrichit la version de 1911 et compte 30 000 mots.

b) Princeton WordNet et autres wordnets

WordNet, développé depuis 1985 à l'Université de Princeton, constitue un réseau sémantique à large couverture de la langue anglaise (206 941 lexies décrivant 117 659 concepts dans la version 3.0). Les entrées y sont structurées par un ensemble riche de relations lexicales et sémantiques.

Plusieurs projets ont vu le jour pour créer des wordnets dans d'autres langues. On peut notamment citer WOLF pour le français. Toutefois, aucun de ces wordnets n'atteint pour l'instant la largeur de couverture de la version anglaise.

c) Wikipédias

Lancée en 2001, l'encyclopédie libre collaborative Wikipédia compte en juin 2011 plus de 18 millions d'articles en 281 versions (et presque autant de langues), avec 37 langues dotées de plus de 100 000 articles. Ses intérêts en TAL sont multiples. Elle peut être vue comme un corpus multilingue de

[66] De plus, on peut imaginer que WordNet n'est cité que dans le cadre d'articles autour du TAL alors que Wikipédia l'est aussi dans d'autres contextes.

volume significatif, qui permet de réaliser des comptages statistiques dans un grand nombre de langues, certaines d'entre elles étant faiblement dotées en ressources lexicales.

Wikipédia propose, en plus du texte encyclopédique, un premier niveau de structuration des connaissances ; la limite pratique est la bonne volonté (ou la compétence) des internautes qui éditent les articles. La figure 10 illustre ces différentes possibilités de structuration :

- Le texte encyclopédique (point 1 sur la figure) est structuré en sections et sous-sections.
- Les « InfoBox » sont des tables préformatées présentant des données importantes sur un sujet, sous forme d'un encadré placé en haut à droite (2) ou à la fin (3) de l'article.
- Les articles peuvent être rattachés à des portails (4), c'est-à-dire des regroupements thématiques permettant de se repérer plus facilement.
- Un article est classé dans une ou plusieurs catégories, présentes en bas de chaque page (5) ; les catégories forment un système de classement thématique organisé selon un graphe orienté[67].
- Les articles en différentes langues, portant sur le même sujet, sont reliés entre eux par l'intermédiaire d'un index interlingue affiché à gauche (6).

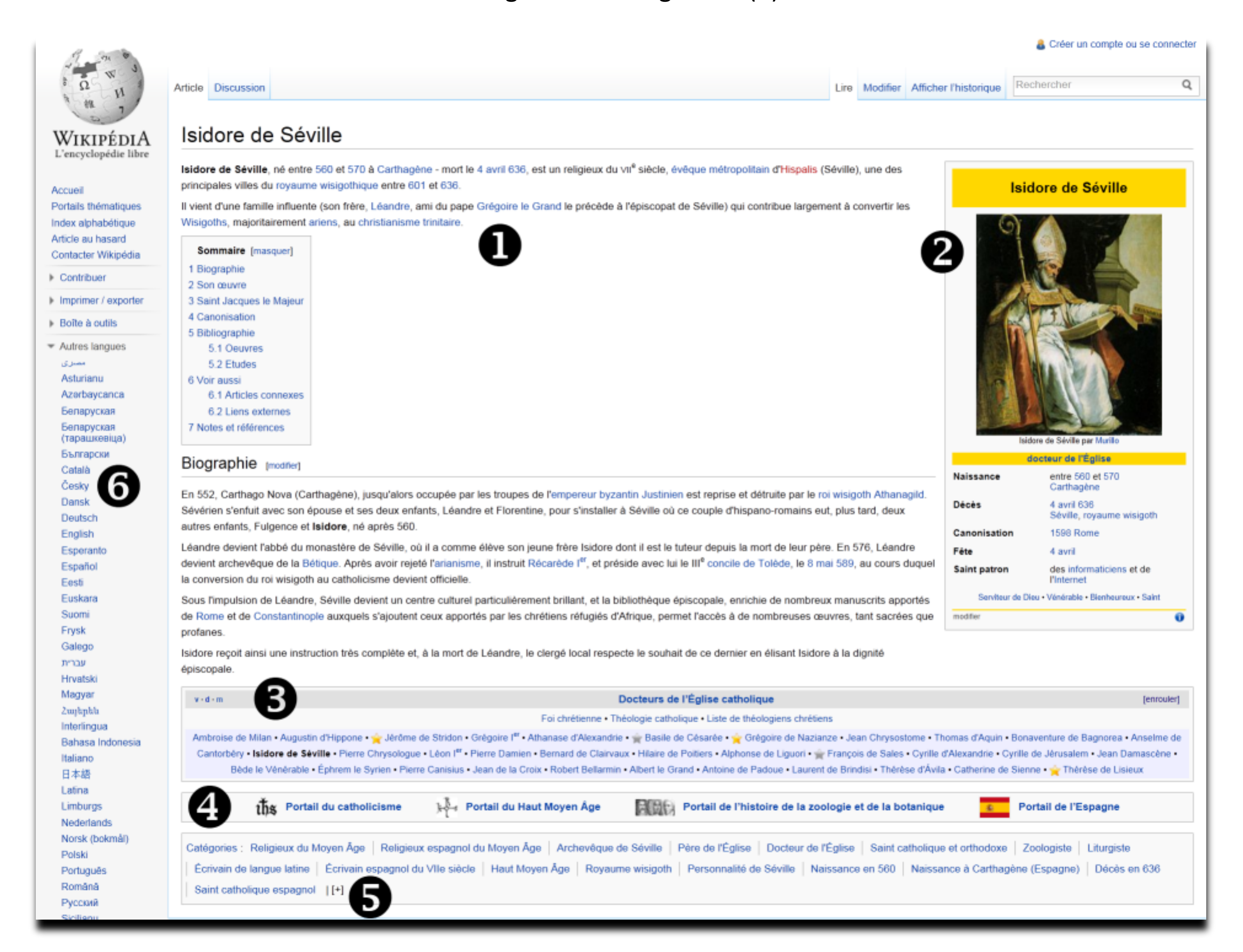


Figure 10 : Exemple de page de la Wikipédia française (article sur saint Isidore)

[67] En pratique, dans la Wikipédia française et l'anglaise, ce graphe comporte des cycles, ce qui en complexifie l'exploration par un algorithme ; en effet, il faut détecter les cycles préalablement à un traitement, ce qui n'est pas chose aisée dans un graphe de plusieurs dizaines de milliers de nœuds.

d) Ressources produites à partir des données structurées de la Wikipédia

DBpedia (Auer *et al.*, 2007) est un projet collaboratif, qui vise à extraire des informations structurées à partir des InfoBox de la Wikipédia et à les rendre disponibles dans les formats et protocoles du Web sémantique, sous forme de triplets RDF[68]. DBpedia permet donc de faire des requêtes complexes[69] sur les connaissances contenues dans la Wikipédia en les reliant à d'autres jeux de données. FreeBase[70] (Bollacker *et al.*, 2008) vise les mêmes objectifs.

YAGO (Suchanek *et al.*, 2007) exploite le graphe de catégories de la Wikipédia et WordNet pour créer automatiquement une vaste ontologie du sens commun (*Yet Another Great Ontology*). La version 2 utilise aussi la base de données géographiques GeoNames et met l'accent sur la qualité des données spatiales et temporelles associées aux faits extraits de la Wikipédia. La précision de cette ontologie construite automatiquement est évaluée par ses auteurs à 95 %.

e) Ontologies

D'autres ontologies proposent une axiomatique plus ou moins riche. On peut notamment citer SUMO (qui vise à proposer un « haut d'ontologie » universel), DOLCE et CYC (dont l'ambition est de décrire très largement le sens commun). COSMO est une proposition de fusion au format OWL des hauts d'ontologie de CYC, SUMO et DOLCE.

Il nous semble que ces ontologies sont plus utilisées dans des travaux relevant de l'intelligence artificielle que du TAL, même si la limite entre ces deux disciplines est parfois floue. Certaines de ces ressources ont un lien explicite vers des entrées de WordNet ou de la Wikipédia anglaise.

3. Contribution des standards émergents du Web sémantique

Le Web sémantique est une évolution du Web classique qui vise à rendre les données accessibles, non seulement à un œil humain, mais aussi aux machines. Pour donner une analogie, pensons à une boîte dans un rayon de supermarché : pour en connaître le contenu, un humain lira le nom du produit sur la boîte ; en revanche, l'ordinateur de la caisse en lira le code-barres et se servira de cette information pour automatiser des traitements (facturation, mise à jour du stock, réapprovisionnement). En offrant cette dualité, le Web sémantique ambitionne de doter les applications de capacités de raisonnement ; le corollaire sera la possibilité d'automatiser des tâches aujourd'hui réservées aux êtres humains, grâce à des agents conversationnels intelligents[71].

---

[68] Dans la version 3.7 datant de septembre 2011, DBpedia compte plus d'un milliard de triplets.

[69] En utilisant SPARQL, le langage de requête du Web sémantique.

[70] Développé initialement par la société MetaWeb, rachetée par Google en juillet 2010 et utilisée depuis mai 2012 sur la version US du moteur de recherche pour alimenter un *Knowledge Graph*.

[71] (Berners-Lee *et al.*, 2001) finissait sur la promesse de tels agents logiciels : *"The real power of the Semantic Web will be realized when people create many programs that collect Web content from diverse sources, process the information and exchange the results with other programs. The effectiveness of such software agents will increase exponentially as more machine-readable Web content and automated services (including other agents) become available."* On peut estimer que cet objectif n'est atteignable qu'à long terme. Néanmoins, l'iPhone 4S lancé en octobre 2011 intègre l'application Siri, qui permet d'utiliser la voix pour (entre autres) envoyer des messages, définir des rappels ou passer des appels téléphoniques. C'est un pas significatif dans la direction d'agents conversationnels grand public, utilisables sans apprentissage préalable.

Dans le survol des ressources proposé à la section précédente, on constate que plusieurs d'entre elles trouvent leurs racines dans le Web sémantique et que plusieurs projets visent à mettre automatiquement en relation différentes ressources. L'apparition récente des formats d'ontologie du Web sémantique est un facteur important de normalisation de ces différentes données. Ils permettent en effet de les représenter d'une façon unifiée[72] ; ces formats autorisent aussi la mise en correspondance entre des concepts identiques, définis dans des référentiels linguistiques différents. Par exemple, on peut facilement exprimer le fait que les concepts DOMESTIC_CAT#1 (dans WordNet 3.0) et CAT (dans DBpedia) sont identiques ; ainsi, on regroupe facilement des connaissances linguistiques (morphologie, hyperonymie…) et encyclopédiques sur le même sujet.

L'annexe 1, page 187, présente en détail le Web sémantique et ses standards émergents. En TAL, ils permettent de représenter non seulement des référentiels linguistiques, mais aussi des graphes complexes (issus de traitements d'analyse syntaxique ou d'extraction d'entités nommées et de relations, par exemple). Une façon rapide d'implémenter une application d'extraction d'information consiste alors à faire une requête SPARQL sur de tels graphes.

B. WordNet et son écosystème

1. Princeton WordNet

Le Princeton WordNet version 3.0 constitue la base du lexique sémantique d'Antelope. Ce projet, mené depuis 1985 à Princeton, offre un réseau sémantique très complet de la langue anglaise. WordNet est utilisable librement, y compris pour un usage commercial, ce qui en a favorisé une diffusion très large. S'il n'est pas exempt de critiques (granularité très fine, absence de certaines relations…), il n'en reste pas moins l'une des ressources de TAL les plus populaires.

a) Notion de synset

WordNet est construit sous la forme d'une hiérarchie de concepts appelés *synsets*[73] qui en forment la composante atomique. Un synset est un ensemble de lexies synonymes entre elles ; un synset correspond donc à un groupe de mots interchangeables, dénotant un sens ou un usage particulier. Un synset est défini d'une façon différentielle par les relations sémantiques (hypéronymie, méronymie, antonymie, etc.) qu'il entretient avec les sens voisins. Dans la suite, nous noterons entre accolades les différentes lexies synonymes qui définissent un synset, sachant que LEXIE#i désigne la i^ème^ lexie d'un vocable dans WordNet. La version 3.0, la plus récente (janvier 2007), compte 117 659 synsets et 206 941 lexies.

Chaque synset est également associé à une définition lexicographique. Nous la préciserons éventuellement en italiques et entre parenthèses après la liste de lexies. Par exemple, le concept « langue naturelle » est défini par le synset suivant :

{NATURAL LANGUAGE#1, TONGUE#2} (*a human written or spoken language used by a community*)

Les **noms** et **verbes** sont organisés en hiérarchies. Des relations d'hyperonymie (« est-un ») et d'hyponymie relient les « ancêtres » des noms et des verbes avec leurs « spécialisations ». Au niveau racine, ces hiérarchies sont organisées en types de base. Le réseau des noms est bien plus profond

[72] On peut par exemple représenter WordNet au format SKOS, voir page 166.

[73] *Synset* est la contraction de *synonym set* (ensemble de synonymes).

que celui des autres parties du discours. À titre indicatif, les deux premiers niveaux de la hiérarchie des noms se composent des concepts abstraits suivants :

- **ABSTRACTION**: ATTRIBUTE, MEASURE/QUANTITY/AMOUNT, RELATION, SET, SPACE, TIME…
- **HUMAN ACTION**: ACTIVITY, COMMUNICATION, DISTRIBUTION, INACTIVITY, JUDGMENT, LEANING, LEGITIMATION, MOTIVATION, PROCLAMATION, PRODUCTION, SPEECH ACT…
- **ENTITY**: ANTICIPATION, CAUSAL AGENT, ENCLOSURE, EXPANSE, LOCATION, PHYSICAL OBJECT, SKY, SUBSTANCE, THING…
- **EVENT**: GROUP ACTION, NATURAL EVENT, MIGHT-HAVE-BEEN, MIGRATION, MIRACLE, NONEVENT, SOCIAL EVENT…
- **GROUP, GROUPING**: ASSOCIATION, BIOLOGICAL GROUP, PEOPLE, COLLECTION, AGGREGATION, COMMUNITY, ETHNIC GROUP, KINGDOM, MULTITUDE, POPULATION, RACE, RARE-EARTH ELEMENT…
- **PHENOMENON**: EFFECT/RESULT, LEVITATION, FORTUNE/CHANCE, REBIRTH, NATURAL PHENOMENON, PROCESS, PULSATION…
- **POSSESSION**: ASSETS, CIRCUMSTANCES, PROPERTY/MATERIAL POSSESSION, TRANSFERRED PROPERTY, TREASURE…
- **PSYCHOLOGICAL FEATURE**: COGNITION/KNOWLEDGE, FEELING, MOTIVATION/NEED…
- **STATE**: ACTION/ACTIVITY, EXISTENCE, STATE OF MIND, CONDITION, CONFLICT, DAMNATION, DEATH, DEGREE, DEPENDENCY, DISORDER, EMPLOYMENT, END, FREEDOM, ANTAGONISM, IMMATURITY, IMMINENCE, IMPERFECTION, INTEGRITY, MATURITY, OMNIPOTENCE, PERFECTION, PHYSIOLOGICAL STATE, RELATIONSHIP, STATE OF AFFAIRS, STATUS, TEMPORARY STATE, NATURAL STATE…

L'organisation des **adjectifs** est différente. Un sens « tête » joue un rôle d'attracteur ; des adjectifs « satellites » lui sont reliés par des relations de synonymie. On a donc une partition de l'ensemble des adjectifs en petits groupes. Les **adverbes** sont le plus souvent définis par les adjectifs dont ils dérivent. Ils héritent donc de la structure des adjectifs.

b) Relations sémantiques (entre synsets)

Le tableau 3 présente un comptage des relations sémantiques de WordNet 2.1 par catégorie.

Relation	Entre…	…et	Nombre	Exemple
Hypernym/Hyponym	Verbe	Verbe	13 124	EXHALE / BREATHE
	Nom	Nom	75 134	FELINE / CAT
Instance Hyponym	Nom	Nom	8 515	EIFFEL TOWER / TOWER
Part	Nom	Nom	8 874	FRANCE / EUROPE
Member	Nom	Nom	12 262	FRANCE / EUROPEAN UNION
Substance	Nom	Nom	793	SERUM / BLOOD
Attribute	Adjectif	Nom	643	INACCURATE / ACCURACY
Verb Group	Verbe	Verbe	1 748	GELATINIZE#1 / GELATINIZE#2
Verb Entailment	Verbe	Verbe	409	DREAM / SLEEP
Verb Cause	Verbe	Verbe	219	ANESTHETIZE / SLEEP
Adjective Similar	Adjectif	Adjectif	22 622	DYING / MORIBUND
Topic Domain	Nom	Adjectif	1 108	COMPUTER SCIENCE / ADDRESSABLE
	Nom	Nom	4 146	COMPUTER SCIENCE / COMPUTER
	Nom	Adverbe	37	
	Nom	Verbe	1 236	COMPUTER SCIENCE / CASCADE

Region Domain	Nom	Adjectif	75	
	Nom	Nom	1 246	FRENCHMAN / FRANCE
Usage Domain	Nom	Adjectif	227	
	Nom	Nom	563	NEUTRALIZATION / EUPHEMISM
	Nom	Adverbe	73	
	Nom	Verbe	14	
See Also	Adjectif	Adjectif	2 683	BLACK / DARK

Tableau 3 : Comptage des relations sémantiques de WordNet

D'autres ressources permettent d'étendre ces relations, voire de créer de nouveaux types de relations. Par exemple, la ressource WordNet Domains (voir page 60) permet d'ajouter de nouvelles instances de relations de type *Topic Domain*, *Region Domain*, *Usage Domain*. On peut aussi enrichir WordNet avec des relations d'un nouveau type ; nous verrons en IV.C.4, page 71, comment nous avons créé d'une façon semi-automatique deux nouvelles catégories de relations concernant les métonymies et les métaphores.

c) Relations lexicales (entre lemmes)

Le tableau 4 présente un comptage des relations lexicales de WordNet 2.1 par catégorie.

Relation	Entre…	…et	Nombre	Exemple
Usage Domain	Nom	Nom	379	
See Also	Verbe	Verbe	582	SLEEP LATE / SLEEP
Adjective Participle	Adjectif	Verbe	124	APPLIED / APPLY
Antonym	Adjective	Adjective	4 080	GOOD / BAD
	Adverbe	Adverbe	718	POORLY / WELL
	Nom	Nom	2 142	WINNER / LOOSER
	Verbe	Verbe	1 089	DIE / BE BORN
Pertainym	Adjectif	Nom	4 814	ACADEMIC / ACADEMIA
	Adverbe	Adjectif	3 213	BOASTFULLY / BOASTFUL
	Adjectif	Adjectif	38	
Derivation	Nom	Verbe	21 579	KILLING / KILL
	Adjectif	Nom	11 401	DARK / DARKNESS
	Nom	Nom	2 931	AUTOMOBILE / AUTOMOBILIST
	Verbe	Adjectif	1 508	KILL / KILLABLE
Adjective Cluster	Adjectif	Adjectif	1 290	STRIDENT / NOISY

Tableau 4 : Comptage des relations lexicales de WordNet

d) Exemples de relations d'hyperonymie et d'hyponymie

Dans l'exemple montré en figure 11 ci-dessous, nous voyons qu'en partant du sens le plus général du nom CAT#1 (chat domestique), on obtient une liste ordonnée d'ancêtres et de descendants, permettant de déterminer qu'un chat est un carnivore, un mammifère, un animal, etc.

Noun cat, true cat

(feline mammal usually having thick soft fur and no ability to roar: domestic cats; wildcats)

Hypernym

feline, felid -- (any of various lithe-bodied roundheaded fissiped mammals, many with retractile claws)
carnivore -- (a terrestrial or aquatic flesh-eating mammal; *terrestrial carnivores have four or five clawed digits on each limb*)
placental, placental mammal, eutherian, eutherian mammal -- (mammals having a placenta; all mammals except monotremes and marsupials)
mammal, mammalian -- (any warm-blooded vertebrate having the skin more or less covered with hair; young are born alive except for the small subclass of monotremes and nourished with milk)
vertebrate, craniate -- (animals having a bony or cartilaginous skeleton with a segmented spinal column and a large brain enclosed in a skull or cranium)
chordate -- (any animal of the phylum Chordata having a notochord or spinal column)
animal, animate being, beast, brute, creature, fauna -- (a living organism characterized by voluntary movement)
organism, being -- (a living thing that has (or can develop) the ability to act or function independently)
living thing, animate thing -- (a living (or once living) entity)
whole, unit -- (an assemblage of parts that is regarded as a single entity; *how big is that part compared to the whole?; the team is a unit*)
object, physical object -- (a tangible and visible entity; an entity that can cast a shadow; *it was full of rackets, balls and other objects*)
physical entity -- (an entity that has physical existence)
entity -- (that which is perceived or known or inferred to have its own distinct existence (living or nonliving))

Hyponym

domestic cat, house cat, Felis domesticus, Felis catus -- (any domesticated member of the genus Felis)
wildcat -- (any small or medium-sized cat resembling the domestic cat and living in the wild)

Figure 11 : Exemple de relations d'hyperonymie et d'hyponymie

Dans WordNet, chaque nom ou verbe peut avoir plusieurs hyperonymes. Par exemple, le nom ADRENALIN a pour hyperonymes : CATECHOLAMINE, HORMONE, NEUROTRANSMITTER, VASOCONSTRICTOR[74].

La possibilité de définir de multiples hyperonymes est relativement peu utilisée[75], néanmoins il est indispensable de la prendre en compte. Nous avons donc développé une « projection à plat » des hyperonymes, en transformant le graphe orienté acyclique d'héritage en une structure linéaire respectant un ordre partiel. Son intérêt est de faciliter l'écriture de certains algorithmes, qui ont besoin d'énumérer les hyperonymes d'un synset, en leur fournissant une liste plutôt qu'un graphe. La figure 12 montre les hyperonymes du synset BREAD#1 'pain' sous forme de graphe (à gauche) et de liste (à droite).

[74] On remarquera que WordNet n'est pas exempt d'erreurs mineures : on voit ici une redondance, CATECHOLAMINE étant déjà un hyponyme de HORMONE.
[75] Dans WordNet version 2.0, sur un total de 92 634 relations d'héritages entre synsets (noms et verbes), seulement 2 117 présentaient un héritage multiple (et seulement 70 avaient 3 hyperonymes directs ou plus).

Noun bread, breadstuff, staff of life

Hypernym

- baked goods
  - food, solid food
    - solid
      - matter
        - physical entity
          - entity
- starches
  - foodstuff, food product
    - food, nutrient
      - substance
        - matter
          - physical entity
            - entity

Hypernym

- starches
- foodstuff, food product
- food, nutrient
- substance
- baked goods
- food, solid food
- solid
- matter
- physical entity
- entity

Figure 12 : Hyperonymes du synset BREAD#1 'pain' sous forme de graphe et de liste

e) Exemples de relations d'holonymie et de méronymie

Comme montré en figure 13, on peut déterminer grâce à ces relations qu'un chat a des pattes, un pelage, une queue…

HasPart

(Inherited from feline, felid)
- paw -- (a clawed foot of an animal especially a quadruped)
  - pad -- (the fleshy cushion-like underside of an animal's foot or of a human's finger)

(Inherited from mammal, mammalian)
- coat, pelage -- (growth of hair or wool or fur covering the body of an animal)
- hair, pilus -- (any of the cylindrical filaments characteristically growing from the epidermis of a mammal; *there is a hair in my soup*)

(Inherited from vertebrate, craniate)
- belly -- (the underpart of the body of certain vertebrates such as snakes or fish)
- caudal appendage -- (tail especially of a mammal posterior to and above the anus)
- digit, dactyl -- (a finger or toe in human beings or corresponding body part in other vertebrates)
  - nail -- (horny plate covering and protecting part of the dorsal surface of the digits)
    - half-moon, lunula, lunule -- (the crescent-shaped area at the base of the human fingernail)
    - matrix -- (the formative tissue at the base of a nail)
  - phalanx -- (any of the bones of the fingers or toes)
- rib, costa -- (any of the 12 pairs of curved arches of bone extending from the spine to or toward the sternum in humans (and similar bones in most vertebrates))
  - costal cartilage -- (the cartilages that connect the sternum and the ends of the ribs; its elasticity allows the chest to move in respiration)
- tail -- (the posterior part of the body of a vertebrate especially when elongated and extending beyond the trunk or main part of the body)
  - dock -- (the solid bony part of the tail of an animal as distinguished from the hair)
- thorax, chest, pectus -- (the part of the human torso between the neck and the diaphragm or the corresponding part in other vertebrates)
  - area of cardiac dullness -- (a triangular area of the front of the chest (determined by percussion); corresponds to the part of the heart not covered by the lungs)
  - breast, chest -- (the front of the trunk from the neck to the abdomen; *he beat his breast in anger*)
  - chest cavity, thoracic cavity -- (the cavity in the vertebrate body enclosed by the ribs between the diaphragm and the neck and containing the lungs and heart)

Figure 13 : Exemples de relations d'holonymie et de méronymie

La version 2.1 a introduit la notion d' « instance hyponyme », qui désigne une instance (et non une sous-classe) d'un synset (une entité nommée). Par exemple, GEORGE WASHINGTON est une instance

hyponyme de PRESIDENT OF THE UNITED STATES. De même, le nom TOWER#1 a pour hyponymes SILO, MINARET, PYLON… et TOUR EIFFEL comme instance hyponyme.

f) Notre modélisation de WordNet

La figure 14 présente la modélisation de ce lexique sémantique. Un synset est associé à un ou plusieurs lemmes. Chaque lemme est associé à un seul synset. Un synset est en relation sémantique avec d'autres synsets ; de même, un lemme est en relation lexicale avec d'autres lemmes. Comme une opération fréquemment utilisée dans les algorithmes de parcours du graphe est l'énumération des hyperonymes d'un synset, il les présente aussi sous forme de liste ordonnée.

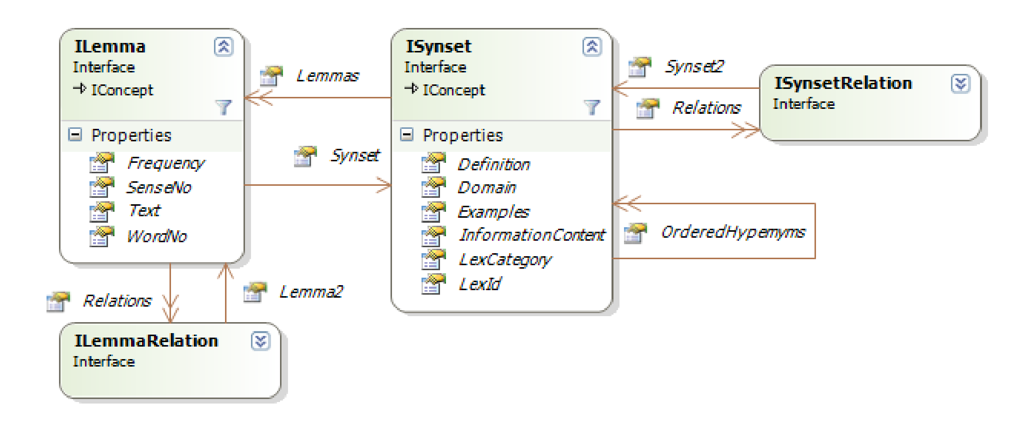


Figure 14 : Modélisation du lexique sémantique

g) Fréquence des lemmes

WordNet associe à chaque lemme une fréquence, qui est son nombre d'occurrences dans le corpus Brown. WordNet peut donc classer les différents sens d'un même mot (les différentes lexies d'un vocable) dans l'ordre décroissant de fréquence.

Si la lexie n'apparaît pas dans le corpus Brown, le lexicographe crée les entrées en fonction de l'importance d'usage supposée. Par exemple, les différents sens de MINK ('vison') sont classés dans l'ordre suivant : 1) fourrure de l'animal ; 2) manteau de fourrure ; 3) animal.

h) Contenu informationnel

Pour un nom ou un verbe, la somme cumulée des fréquences d'un synset et de ses hyponymes au sein d'un sous-arbre de la hiérarchie permet de définir son contenu informationnel. Cette notion donne l'importance relative d'un concept, même si ses lexies apparaissent peu fréquemment dans un corpus.

Par exemple, la fréquence d'apparition de {MAMMAL#1, MAMMALIAN#1} 'mammifère' est très faible (le terme n'apparaît que 3 fois tel quel dans le corpus Brown). En revanche, le concept est important et son contenu informationnel vaut 2 293, correspondant à la somme récursive des fréquences d'apparition de tous ses hyponymes directs ou indirects : CAT#1 'chat' (18), MOUSE#1 'souris' (14), RAT#1 (5), LION#1 (2), etc.

i) Limites de WordNet

WordNet ne donne pas certaines informations usuellement présentes dans un lexique. Par exemple, WordNet ne précise ni l'étymologie ni la prononciation des mots et ne contient que des informations

limitées sur leur usage. Il manque aussi des informations sur la cooccurrence lexicale restreinte (absence de fonctions lexicales).

WordNet propose parfois une profusion de sens pour un mot donné. La contrepartie de son importante couverture est que WordNet est très précis dans le sens des définitions. On a une granularité très (trop ?) fine des sens. Par exemple, le verbe TO GIVE (« donner ») n'a pas moins de 44 sens ; certains de ces sens sont des valeurs de fonctions lexicales et devraient être distingués en tant que tels. Une telle profusion ne facilite pas une tâche de désambiguïsation lexicale.

WordNet manque de relations pragmatiques. En effet, WordNet ne matérialise pas d'une façon formelle tout le sens contenu dans les définitions des termes. Par exemple, l'information « un chat ne rugit pas » figure dans la définition textuelle, mais ne se retrouve formalisée dans aucune relation. De même, des relations qui pourraient exister (comme celle entre SOAP#1 'savon' et BATH#2 'bain', ou celle entre KITTEN#1 'chaton' et CAT#1 'chat') sont absentes de WordNet.

j) Correspondance entre différentes versions

Il existe une correspondance des identifiants de synsets entre versions de WordNet. Elle est indispensable pour assurer une traçabilité avec la version la plus récente. En effet, plusieurs ressources complémentaires à WordNet, et dignes d'intérêt, ont été définies pour d'anciennes versions (1.7 ou 2.0).

Curieusement, le site Web de Princeton n'offre de correspondance « officielle » que pour les noms et les verbes. Heureusement, d'autres sites en proposent également (construites automatiquement) pour les adjectifs et adverbes.

k) Corpus étiquetés par rapport à WordNet

À notre connaissance, peu de corpus sont étiquetés manuellement par rapport aux sens de WordNet. Nous pouvons citer le corpus SemCor (un sous-ensemble du corpus Brown), composé de 352 documents, comptant 2000 mots chacun approximativement.

Plus précisément, le corpus SemCor compte au total 676 546 mots (hors ponctuations). 234 135 noms, verbes, adjectifs et adverbes ont fait l'objet d'une désambiguïsation lexicale manuelle par rapport à WordNet 1.6, puis d'une correspondance automatique vers les versions suivantes de WordNet (jusqu'à la 2.1). Ce corpus permet par exemple un début d'apprentissage automatique pour des tâches de désambiguïsation lexicale.

l) Utilisation dans le cadre de nos travaux

WordNet est notre point de départ pour alimenter un lexique utilisable par la machine. Ce lexique est utilisé :

- D'une part pour déterminer les différents sens d'un mot donné.
- D'autre part pour rechercher quels sens d'un nom vérifient des contraintes de sélection (par exemple, les sens du nom *chat* correspondent à un <animal>).

m) Écosystème de WordNet

Plusieurs autres ressources linguistiques à large couverture, constituées manuellement ou automatiquement, se rattachent à WordNet. Des programmes issus du monde de l'intelligence artificielle ont également établi des passerelles avec WordNet. L'ensemble constitue un écosystème

complet couvrant des aspects lexicaux, syntaxiques et sémantiques. La figure 15 présente quelques-unes de ces ressources. On pourrait y ajouter les ressources que nous avons produites, par exemple les relations de polysémie régulière que nous avons extraites de WordNet (voir page 71).

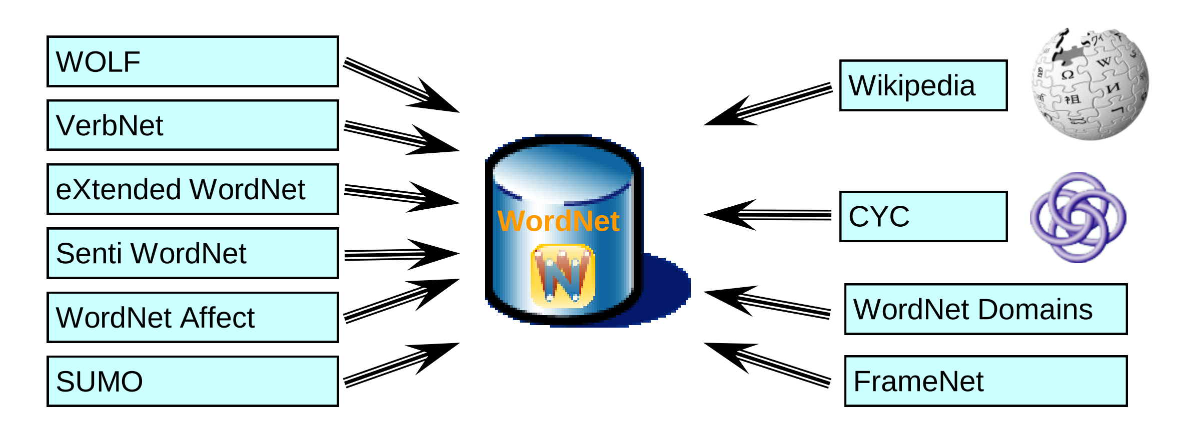


Figure 15 : Liste (non exhaustive) de ressources disposant d'un lien vers WordNet

Antelope combine WordNet avec WOLF (p. 55), eXtended WordNet (p. 55), VerbNet (voir p. 56), WordNet Domains (p. 60), WordNet-Affect (p. 61), SentiWordNet (p. 61), l'ontologie SUMO (p. 62) et une correspondance partielle vers la Wikipédia (p. 64). L'ensemble constitue un lexique sémantique homogène, utilisé pour la désambiguïsation, la résolution d'anaphores et l'ISS.

Notons que l'interopérabilité entre ces différentes ressources lexico-sémantiques et WordNet est permise par la présence d'un identifiant unique pour chaque synset[76]. Ces ressources (à l'exception de la correspondance vers la Wikipédia) fournissent explicitement l'information de traçabilité vers un synset et sont donc homogènes avec WordNet. Leur intégration nécessite un travail d'ingénierie, mais ne soulève pas de difficulté conceptuelle.

En l'absence d'une telle information, l'intégration de la ressource à WordNet se heurte à un problème d'hétérogénéité conceptuelle. Il faut alors établir une correspondance entre chaque entrée d'une telle ressource et un synset ; c'est ce que nous avons établi dans le cas de la Wikipédia (voir le chapitre IV.C).

Combinées, ces ressources fournissent un lexique prêt à l'emploi pour des applications de TAL telles que la recherche d'information, l'inférence pour la compréhension automatique de textes, la désambiguïsation lexicale ou la résolution d'anaphores. Le fait de mettre en commun plusieurs ressources à large couverture permet d'espérer des progrès dans les applications de TAL. Par exemple, (Shi, Mihalcea, 2005) revendique la construction d'un analyseur sémantique robuste en langue anglaise, en utilisant WordNet, VerbNet et FrameNet.

[76] Cet identifiant change hélas à chaque version de WordNet. Pour importer une ressource donnée, il faut donc connaître la version de référence, et utiliser une table de correspondance. L'Université de Catalogne (www.lsi.upc.es/~nlp) propose de telles tables de correspondance.

n) Wordnets pour des langues autres que l'anglais

EuroWordNet est une base de données pour plusieurs langues européennes. La phase initiale du projet s'est achevée en 1999 avec la conception de la base de données, ainsi que la définition de types de relations, d'un haut d'ontologie (63 éléments partagés par toutes les langues) et d'un index inter-langues, en partant de la version 1.5 du WordNet de Princeton. EuroWordNet a produit des wordnets pour le néerlandais, l'italien, l'espagnol, l'allemand, le français, le tchèque et l'estonien[77], comme indiqué dans le tableau 5.

Langue	Synsets	Sens de mots	Relations internes à une langue	Relations d'équivalence entre langues différentes
WordNet 1.5	94 515	187 602	211 375	0
Ajouts à l'anglais	16 361	40 588	42 140	0
Néerlandais	44 015	70 201	111 639	53 448
Espagnol	23 370	50 526	55 163	21 236
Italien	40 428	48 499	117 068	71 789
Allemand	15 132	20 453	34 818	16 347
Français	22 745	32 809	49 494	22 730
Tchèque	12 824	19 949	26 259	12 824
Estonien	7 678	13 839	16 318	9 004

Tableau 5 : Langues proposées dans EuroWordNet

Les langues sont reliées par l'intermédiaire d'un index inter-langues. Il est ainsi possible de passer des mots dans une langue aux mêmes mots dans n'importe quelle autre langue. EuroWordNet permet donc en principe une recherche d'information monolingue ou multilingue.

On peut regretter qu'EuroWordNet ne soit pas distribué librement, contrairement à la version de Princeton. Cela explique certainement sa diffusion beaucoup moins importante.

Plusieurs autres groupes de recherche ont développé des wordnets dans d'autres langues en se basant sur les spécifications d'EuroWordNet (suédois, norvégien, danois, grec, portugais, basque, catalan, roumain, lithuanien, russe, bulgare et slovène). Un autre projet, BalkaNet, prolonge la base de données d'EuroWordNet avec d'autres langues européennes et fournit dans un format XML des ressources pour le tchèque, le roumain, le grec, le turc, le bulgare, et le serbe, comme il est montré dans le tableau 6.

	Bulgare	Tchèque	Grec	Roumain	Turc	Serbe
Synsets	21 441	28 456	18 461	19 839	14 626	8 059
Noms	14 174	21 009	14 426	13 345	11 059	5 919
Verbes	4 169	5 155	3 402	4 808	2 725	1 803
Adjectifs	3 088	2 128	617	852	802	324
Adverbes	9	164	16	834	40	13
Lemmes	44 956	43 918	24 366	33 690	20 310	13 295

Tableau 6 : Langues proposées dans BalkaNet

[77] À notre connaissance, les ressources pour le français ont été fournies par la société MemoData sur la base de son Dictionnaire Intégral.

o) WOLF

WOLF (Sagot, Fišer, 2008) est un WordNet libre du français construit à partir du Princeton WordNet et de diverses ressources multilingues. Les lexèmes polysémiques ont été traités par alignement d'un corpus parallèle en cinq langues ; le lexique multilingue extrait a été désambiguïsé sémantiquement à l'aide des wordnets des langues concernées. Une approche bilingue, obtenue à partir de la Wikipédia et de thésaurus, a permis de construire de nouvelles entrées grâce aux mots monosémiques.

2. Gloses désambiguïsées

a) eXtended WordNet

Mené à l'Université de Dallas, eXtended WordNet –ou XWN– (Mihalcea, Moldovan, 2001) enrichit WordNet 2.0 en associant à chaque synset une analyse syntaxique de sa définition, la désambiguïsation lexicale de chaque mot de la définition, ainsi qu'une forme logique. Par exemple, la définition du nom COUSIN#1, "*the child of your aunt or uncle*" (« l'enfant de votre tante ou de votre oncle »), a pour analyse syntaxique :

```
(TOP (S (NP (NN cousin) )
        (VP (VBZ is)
                (NP (NP (DT the) (NN child) )
                        (PP (IN of)
                                (NP (PRP$ your) (NN aunt) (CC or) (NN uncle) ) ) ) )
(. .) ) )
```

Ainsi que la forme logique suivante :

```
cousin:NN(x1) -> child:NN(x1) of:IN(x1, x4) aunt:NN(x2) or:CC(x4, x2, x3) uncle:NN(x3)
```

Les informations présentes dans XWN sont de qualité *gold* (validé humainement), *silver* (accord entre deux analyseurs syntaxiques) ou *normal*. Le tableau 6 présente le taux de validation des mots des définitions dans cette ressource, par partie du discours.

Synsets (WN 2.0)	Nombre de définitions	Mots de classe ouverte	Mots mono sémiques	Qualité *gold*	Qualité *silver*	Qualité *normal*
Noms	79 689	505 946	138 274	10 142	45 015	296 045
Verbes	13 508	48 200	6 903	2 212	5 193	30 813
Adjectifs	18 563	74 108	14 142	263	6 599	50 359
Adverbes	3 664	8 998	1 605	1 829	385	4 920

Tableau 7 : Taux de validation des mots des définitions dans eXtended WordNet

Pour un total de 637 252 mots de classe ouverte utilisés dans les définitions, seuls 14 446 mots sont de qualité *gold* (2,3 %). Du fait de la complexité de la tâche de désambiguïsation lexicale, et de l'absence de validation humaine systématique, il est sage de penser que les mots étiquetés avec une qualité *silver* ou *normal* ne sont pas forcément désambiguïsés d'une façon correcte (les définitions *gold* ne représentent que 3,2 % des mots polysémiques).

En dépit de ces limitations, XWN a été utilisé pour améliorer les résultats d'un système de questions-réponses (Moldovan, Novischi, 2002).

b) *WordNet Gloss Corpus*

Le lexique d'Antelope a finalement remplacé XWN par les données du *WordNet Gloss Corpus*. Publié en avril 2008 par l'Université de Princeton, ce projet offre le double avantage de reposer sur WordNet version 3.0 et de disposer d'une validation beaucoup plus couvrante (29 % des mots ont été désambiguïsés manuellement).

c) *Utilisation dans le cadre de nos travaux*

Les gloses désambiguïsées nous servent, dans le cadre de l'analyse sémantique, à déterminer les contraintes de sélection. En effet, elles nous permettent de savoir si un nom a un trait particulier (<rigide>, <allongé>, <pointu>…). Pour ce faire, nous recherchons un adjectif de ce type dans les mots de la définition du nom ou de ses hyperonymes.

3. Cadres de sous-catégorisation des verbes

Connaître les cadres de sous-catégorisation des verbes est un élément essentiel d'une ISS. Ils peuvent provenir de différentes ressources proposant des informations de contrainte de sélection plus ou moins fines.

Des dictionnaires généralistes offrent souvent un premier niveau, grossier, d'informations de ce type. Par exemple, dans WordNet, chaque verbe est associé à un ou éventuellement plusieurs cadres donnant un premier niveau de typage : il existe une dizaine de cadres en tout : *Somebody ----s something*, *Somebody ----s*, *Somebody ----s PP*… sans réelle explicitation des contraintes de sélection.

Certaines ressources dédiées sont construites manuellement. Les plus connues pour la langue anglaise sont VerbNet (présentée en détail dans cette section) et FrameNet (que nous survolerons page 83). VerbNet s'appuie sur une vingtaine de rôles thématiques et une quarantaine de contraintes de sélection.

Pour les verbes français, la ressource la plus complète semble être le Lexique-Grammaire du LADL (Gross, 1994). Des lexiques comme Dicovalence[78] (van den Eynde, Mertens, 2003), le DEV (Dictionnaire Electronique des Verbes français) de Dubois, ou le Le*fff* (Clément, Sagot, Lang, 2004) apportent aussi des descriptions fines des cadres de sous-catégorisation pour le français. (Danlos, Sagot, 2007) compare les modèles lexicaux du Lexique-Grammaire, de Dicovalence et du Le*fff*.

D'autres ressources résultent d'un mécanisme d'apprentissage. Par exemple, (Messiant, Gábor, Poibeau, 2010) décrit une méthode permettant l'acquisition automatique d'un lexique de sous-catégorisation des verbes français (LexSchem), à partir de l'analyse syntaxique du corpus LM10[79]. Nous montrerons page 68 comment l'apprentissage de paraphrases (obtenues à partir de paires d'articles encyclopédiques comparables) nous permet de construire des cadres de sous-catégorisation fins, dont les actants sont désambiguïsés par rapport à WordNet, comme par exemple : SERPENTER#1 (RIVIERE#1, VILLE#1) ~ COULER#2 (RIVIERE#1, VILLE#1).

a) *Présentation de VerbNet*

Nous avons intégré dans le lexique d'Antelope un lexique des classes de verbes anglais, VerbNet. Mené sous l'impulsion de Martha Palmer (d'abord à l'Université de Pennsylvanie, puis à Boulder au

[78] Dicovalence, fondé sur l'approche pronominale, donne les cadres valenciels de 3 700 verbes.
[79] Un corpus journalistique, obtenu par analyse syntaxique des articles de 10 années du journal Le Monde.

Colorado), VerbNet regroupe par classe les verbes partageant les mêmes comportements syntaxiques et sémantiques. C'est un prolongement des travaux de (Levin, 1993).

Une classe de verbes regroupe plusieurs verbes et identifie des rôles thématiques avec d'éventuelles contraintes de sélection. Elle décrit plusieurs constructions typiques (des *frames* en anglais) des verbes membres. La sémantique de l'action ou de l'événement est également précisée. Des sous-classes permettent de décrire d'éventuelles spécialisations d'une classe. On peut en trouver une description dans (Kipper-Schuler, 2003).

La version 2.1 distingue 237 classes de verbes qui regroupent 4991 sens de verbes. Un verbe membre d'une classe est souvent accompagné d'une précision sur le synset correspondant, qui permet d'identifier dans WordNet le sens précis du verbe. VerbNet dispose aussi d'une correspondance vers FrameNet. Chaque fichier de VerbNet décrivant une classe de verbes est représenté en XML et découpé en sections balisées selon une structure arborescente :

- <MEMBERS> décrit les verbes membres qui appartiennent à la classe, en précisant l'identifiant vers les synsets correspondants de WordNet.
- <THEMROLES> indique les rôles thématiques de la classe :
    - <SELRESTRS> précise leurs éventuelles contraintes de sélections.
- <FRAMES> indique chacune des constructions typiques en donnant à chaque fois :
    - <SYNTAX> sa syntaxe.
    - <SEMANTICS> sa sémantique.
    - <EXAMPLES> un ou plusieurs exemples.
- <SUBCLASSES> regroupe éventuellement en sous-classes :
    - <VNSUBCLASS> les cas particulier d'une classe de verbes.

b) Les rôles thématiques

Les rôles thématiques font référence aux relations sémantiques sous-jacentes entre un prédicat et ses arguments. Ils ont été introduits à la fin des années 60 (Gruber, 1965 ; Fillmore, 1968 ; Jackendoff, 1972) de façon à créer un ensemble fini de types de participants en tant qu'arguments de prédicat. Ces rôles sont utilisés pour décrire les comportements lexicaux et syntaxiques des verbes.

Ces rôles sont indépendants de la construction syntaxique. Par exemple, dans les deux phrases suivantes, « Jean » a le rôle thématique *Patient* de l'action de frapper, et «Marie » a le rôle *Agent* :

- Marie frappe Jean.
- Jean est frappé par Marie.

Chaque argument du verbe (chaque actant) joue un rôle thématique. Il peut être, par exemple, *Agent*, *Patient*, *Thème*, *Instrument*, *Source*… de l'action ou de l'événement décrit par le verbe. Chaque argument d'un verbe est assigné à un unique rôle thématique au sein d'une classe de verbe. L'une des exceptions à cette règle concerne les classes contenant des verbes avec des arguments symétriques (comme dans « Jean et Marie discutent » ou « La France et l'Italie se touchent ») qui ont alors deux arguments (ou plus) tels qu'*Acteur1* et *Acteur2*, mais du même type.

VerbNet définit une vingtaine de rôles thématiques. Ils sont énumérés en annexe, page 209, avec pour chaque rôle un exemple à titre d'illustration.

c) Contraintes de sélection

Un rôle thématique peut avoir des contraintes de sélection, qui en restreignent les sens possibles. Un *Agent* a généralement une contrainte de sélection <humain> ou <animé>. VerbNet en propose une quarantaine, organisées selon un graphe d'héritage, comme le montre la figure 16.

L'un des enjeux de l'ISS, lors de la désambiguïsation lexicale, est d'établir une correspondance entre les mots du lexique et la hiérarchie des contraintes de sélection.

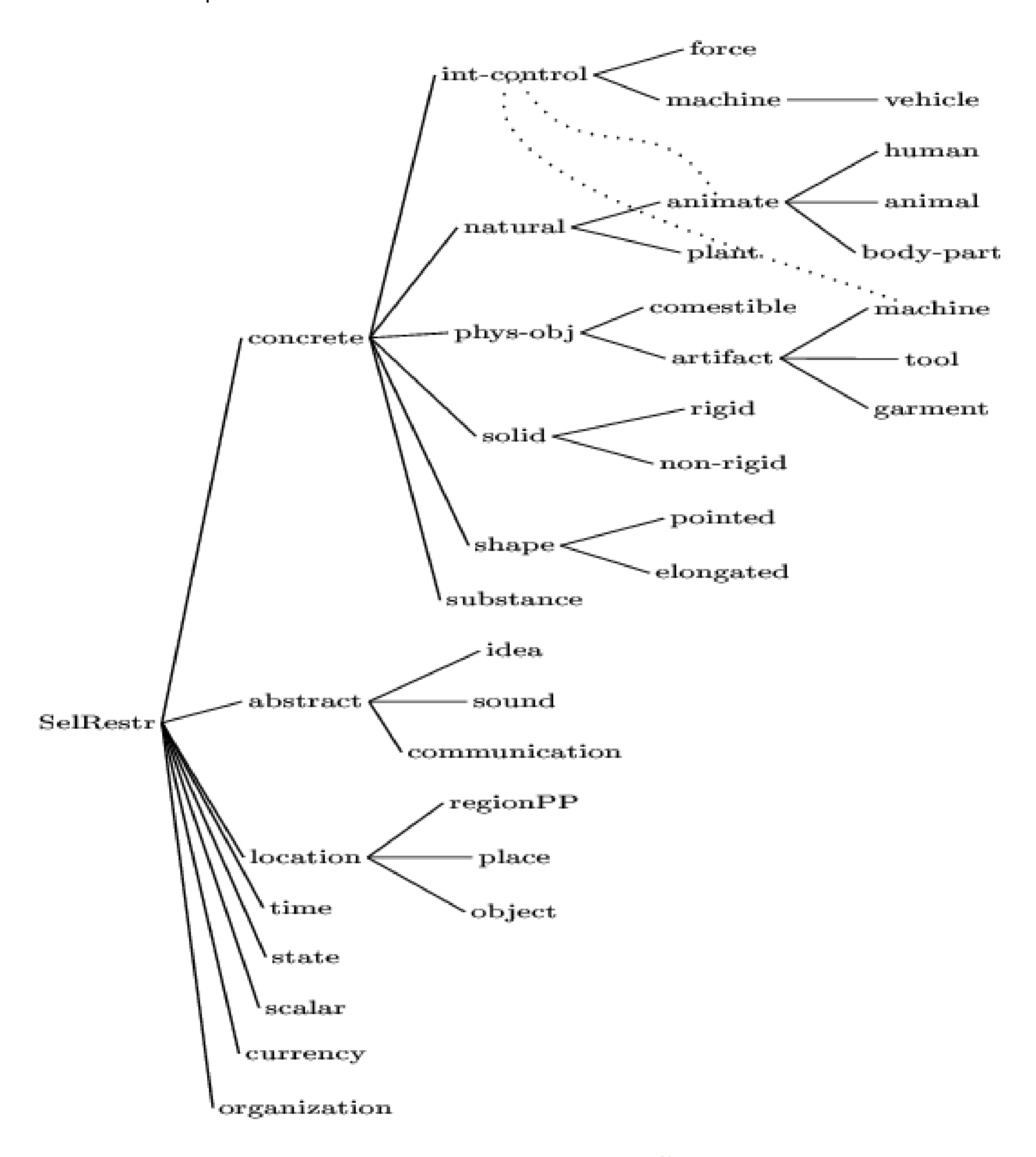


Figure 16 : Hiérarchie des contraintes de sélection définies par VerbNet[80]

d) Exemple : la classe de verbe "murder"

Illustrons cette structure sur un exemple, la classe de verbe "murder". Le fichier `murder.xml` décrit trois constructions typiques :

[80] Source (Kipper-Schuler, 2003).

- *Agent* élimine *Patient* (« Brutus tua Jules César »).
- *Agent* élimine *Patient* avec *Instrument* (« Brutus tua César avec un poignard »).
- *Instrument* élimine *Patient* (« le pesticide tua les insectes »)[81].

Chaque description de classe de verbes déclare des contraintes de sélection sur les rôles thématiques. Par exemple, pour “**murder**”, l’*Agent* et le *Patient* doivent avoir un trait <animé> (en pratique, <humain> ou <organisation>) et l’*Instrument* doit être <concret>.

Le fichier précise ensuite la syntaxe et la sémantique de la classe de verbe. Par exemple, la deuxième construction de la classe de verbe “**murder**” décrit :

```
<SYNTAX>
      <NP value="Agent"/>
      <VERB/>
      <NP value="Patient"/>
      <PREP value="with"/>
      <NP value="Instrument"/>
</SYNTAX>
<EXAMPLES>
      <EXAMPLE>"Brutus killed Caesar with a knife"</EXAMPLE>
</EXAMPLES>
```

Enfin, sa sémantique est décrite pour préciser qu’au démarrage de l’événement, *Patient* est vivant, mais qu’il ne l’est plus à la fin de l’événement :

- *alive(start(E), Patient)*.
- *! alive(result(E), Patient)*.

e) Prise en compte de l’héritage entre classes

La balise <SUBCLASSES> déclare les éventuelles sous-classes qui spécialisent une classe de verbe donnée. Une sous-classe permet :

- De raffiner les contraintes de sélection portant sur les rôles thématiques.
- De déclarer de nouveaux rôles thématiques.
- D’associer de nouveaux lemmes de WordNet à la sous-classe.
- De créer de nouvelles constructions typiques.

f) Utilisation dans le cadre de nos travaux

Nous présenterons en section V.C.3 (page 110) la façon dont nous utilisons VerbNet (et WordNet) pour implémenter un composant d’étiquetage de rôles thématiques, qui est utilisé au sein de l’ISS. Dans une étape préparatoire, nous traduisons les descriptions en XML des classes de verbes de VerbNet en graphes élémentaires. Lors de l’analyse effective d’un texte, on cherche alors à reconnaître dans le graphe syntaxique (issu d’une analyse en dépendances de chaque phrase) les cadres de sous-catégorisation de verbes, en y recherchant les sous-graphes précompilés lors de l’étape préparatoire.

---

[81] Remarquons que cette entrée est sujette à caution ; seuls les moyens (comme le pesticide) peuvent devenir sujet, mais pas les instruments (**le poignard tua César*).

4. Appartenance d'un synset à un ou plusieurs domaines

a) WordNet Domains

La notion de domaine a été employée aussi bien en linguistique qu'en lexicographie pour marquer des usages des mots. Les domaines sémantiques offrent une manière naturelle d'établir des relations sémantiques entre les sens des mots, qui peuvent être utilisées avec profit en informatique linguistique. WordNet associe parfois explicitement un domaine (*Baseball*, *Géologie*, *Mathématiques*…) à un synset ; toutefois, cette association n'est pas systématique.

WordNet Domains (Magnini, Cavaglià, 2000) est une extension multilingue de WordNet 2.0, développée à l'*Instituto Trentino di Cultura* (ITC-irst). Dans WordNet Domains, chaque synset est annoté avec au moins une étiquette de domaine (par exemple *Sport*, *Politique*, *Médecine*, *Economie*…), choisie dans un ensemble d'environ deux cents étiquettes organisées hiérarchiquement.

Un domaine peut inclure des synsets de différentes parties du discours et de différentes sous-hiérarchies de WordNet. Par exemple le domaine Médecine regroupe des sens de noms tels que DOCTOR#1 et HOSPITAL#1, et de verbes comme OPERATE#7.

L'information apportée par ces domaines est complémentaire à celles déjà présentes dans WordNet. Les domaines peuvent créer des regroupements homogènes des sens d'un même mot, avec comme effet secondaire de réduire la polysémie des mots dans WordNet. L'utilisation de WordNet Domains permet par exemple d'améliorer l'efficacité d'algorithmes de désambiguïsation lexicale et d'expansion de requêtes.

b) Exemple

Le nom BANK, par exemple, a dix sens dans WordNet 2.0. Trois d'entre eux (BANK#1, BANK#3 et BANK#6) sont regroupés au sein du domaine *Economie*, tandis que deux (BANK#2 et BANK#7) sont regroupés avec les étiquettes de domaine *Géographie* et *Géologie*, comme indiqué dans le tableau 8.

Sens	Synset (Définition)	Domaines
#1	{DEPOSITORY FINANCIAL INSTITUTION, BANK#1, BANKING CONCERN, BANKING COMPANY} (*a financial institution…*)	*Economy*
#2	{BANK#2} (*sloping land…*)	*Geography, Geology*
#3	{BANK#3} (*a supply or stock held in reserve…*)	*Economy*
#4	{BANK#4, BANK BUILDING} (*a building…*)	*Architecture, Economy*
#5	{BANK#5} (*an arrangement of similar objects…*)	*Factotum*
#6	{SAVINGS BANK, COIN BANK, MONEY BOX, BANK#6} (*a container…*)	*Economy*
#7	{BANK#7} (*a long ridge or pile…*)	*Geography, Geology*
#8	{BANK#8} (*the funds held by a gambling house…*)	*Economy, Play*
#9	{BANK#9, CANT, CAMBER} (*a slope in the turn of a road…*)	*Architecture*
#10	{BANK#10} (*a flight maneuver…*)	*Transport*

Tableau 8 : Domaines associés aux différents sens du nom BANK

5. Ressources pour l'analyse de sentiments

WordNet-Affect et SentiWordNet sont deux ressources permettent la détection d'affects dans les textes. Elles sont utilisées par l'application d'analyse de sentiments que nous avons développée pour SemEval-2007 (Cf. chapitre V.D). De tels traitements ont un intérêt économique grandissant : par exemple, une société peut chercher à détecter les critiques positives ou négatives sur ses produits en analysant la blogosphère ou les dépêches d'agences de presse.

a) WordNet-Affect

Créé à partir de WordNet Domains, WordNet-Affect (Strapparava, Valitutti, 2004) est une ressource linguistique pour la représentation lexicale de connaissances sur les affects. WordNet-Affect a été développé en deux étapes.

La première a consisté à identifier manuellement un premier « noyau » de synsets affectifs. Un sous-ensemble de synsets de WordNet appropriés est d'abord choisi pour représenter des concepts affectifs ; des informations additionnelles sont ensuite ajoutées aux synsets en leur associant une ou plusieurs étiquettes qui précisent une signification affective. Par exemple, les concepts affectifs représentant un état émotif sont représentés par des synsets marqués par l'étiquette Émotion. Le tableau 9 énumère ces étiquettes affectives, avec des exemples de synsets associés.

La seconde étape a permis, en suivant les relations définies dans WordNet, de propager les informations de ce noyau à son voisinage.

Etiquette affective	Exemples de synsets associés
Emotion	nom ANGER#1, verbe FEAR#1
Mood	nom ANIMOSITY#1, adjectif AMIABLE#1
Trait	nom AGGRESSIVENESS#1, adjectif COMPETITIVE#1
Cognitive State	nom CONFUSION#2, adjectif DAZED#2
Physical State	nom ILLNESS#1, adjectif ALL IN#1
Edonic Signal	nom HURT#3, nom SUFFERING#4
Emotion-Eliciting Situation	nom AWKWARDNESS#3, adjectif OUT OF DANGER#1
Emotional Response	nom COLD SWEAT#1, verbe TREMBLE#2
Behaviour	nom OFFENSE#1, adjectif INHIBITED#1
Attitude	nom INTOLERANCE#1, nom DEFENSIVE#1
Sensation	nom COLDNESS#1, verbe FEEL#3

Tableau 9 : Exemples de synsets associés à des étiquettes affectives

b) SentiWordNet

SentiWordNet (Esuli, Sebastiani, 2006) est une ressource lexicale permettant le sondage d'opinion. SentiWordNet assigne à chaque synset de WordNet 2.0 trois valeurs, la positivité, la négativité et l'objectivité (absence de connotation affective), en respectant l'égalité : positivité + négativité + objectivité = 1. Par exemple, pour les trois sens de l'adjectif "*estimable*", SentiWordNet propose les valences indiquées dans le tableau 10. Le sens « calculable » n'a pas de valence particulière, alors que les deux autres sens sont très positifs.

	P = 0 N = 0 O = 1	{COMPUTABLE#1, **ESTIMABLE**#3} *may be computed or estimated; "a calculable risk"; "computable odds"; "estimable assets"*
	P = 0,75 N = 0 O = 0,25	{**ESTIMABLE**#1} *deserving of respect or high regard*
	P = 0,625 N = 0,25 O = 0,125	{HONORABLE#5, GOOD#4, RESPECTABLE#2, **ESTIMABLE**#2} *deserving of esteem and respect; "all respectable companies give guarantees"; "ruined the family's good name"*

Tableau 10 : Valence affective des trois sens de l'adjectif ESTIMABLE selon SentiWordNet

Cette ressource a été créée d'une façon semi-supervisée, en mixant des règles linguistiques et de l'apprentissage automatique (par utilisation de classifieurs). Les résultats n'ont pas fait l'objet d'une validation manuelle systématique ; certains peuvent sembler incorrects[82].

6. Ontologies SUMO et MILO

SUMO –pour *Suggested Upper Merged Ontology*– (Niles, Pease, 2003) est une proposition de standard soumise à l'IEEE pour représenter un « haut » générique d'ontologie, répertoriant d'une façon réutilisable et générique de grandes catégories de la pensée humaine. MILO (*MId-Level Ontologies*) est un ensemble d'ontologies multi domaines, de niveau intermédiaire, créées en se basant sur SUMO. L'ensemble, écrit en une version simplifiée du *Knowledge Interchange Format* (langage logique du premier ordre), compte 20 000 termes et 60 000 axiomes.

a) Notion de « haut » d'ontologie

Les ontologies sont des artefacts construits en fonction d'une tâche précise. L'une des difficultés généralement constatées est qu'une ontologie donnée est rarement réutilisée pour une tâche autre que celle qui a motivé sa construction originelle. Il découle de ce constat de nombreuses recherches sur la réutilisabilité du « haut » des ontologies ; leur argumentaire est : puisqu'il est difficile, voire impossible, de réutiliser directement des ontologies, trop proches de vues détaillées qu'on peut avoir sur un domaine, intéressons-nous au « haut » de l'ontologie. Cette *Upper Ontology* répertorie et organise de grandes catégories de la pensée ou de la société humaine qui devraient pouvoir être réutilisables dans de très nombreuses applications et être alors génériques.

L'objectif du groupe *Standard Upper Ontology* est de réfléchir à la constitution d'un haut d'ontologie qui se voudrait universel pour les grandes catégories d'objets et de pensées, puis de le soumettre à un processus de normalisation. Le résultat est SUMO, qui cherche à devenir un standard, et commence à être utilisé notamment pour le Web sémantique.

SUMO est écrit en langage SUO-KIF, dérivé simplifié de KIF (*Knowledge Interchange Format*), un langage équivalent à la logique du premier ordre. Une traduction vers OWL (le langage de description

[82] Par exemple, les valeurs associées au nom RAPE#3 'viol' (« *crime consistant à forcer une femme à se soumettre à des rapports sexuels contre sa volonté* ») sont positivité=0,25 et négativité=0 en dépit de la présence du mot « *crime* » dans sa définition.

d'ontologie du Web sémantique) est également disponible ; cette traduction est hélas très partielle d'un point de vue axiomatique, KIF étant d'un pouvoir d'expression plus élevé qu'OWL. Il existe une correspondance complète de SUMO et de MILO vers les différentes versions de WordNet.

b) Exemple : le concept BEVERAGE

Nous allons présenter à titre d'exemple le concept BEVERAGE 'boisson'. La définition lexicographique peut inclure des références (soulignées dans la glose) à d'autres concepts :

Définition : Any food that is ingested by drinking. Note that this class is disjoint with the other subclasses of Food, i.e. Meat and FruitOrVegetable.

La partie taxonomique de SUMO précise les sous-classes :

Sous-classes : Milk, AlcoholicBeverage, Coffee, Tea

Le principal apport de SUMO est de fournir une axiomatique riche. Voici les axiomes associés à BEVERAGE (traduits automatiquement en anglais à partir des expressions en KIF) :

Food is disjointly decomposed into Meat, Beverage

for all beverage *?BEV* holds Liquid is an attribute of *?BEV*

for all drinking *?DRINK* holds if *?BEV* is a patient of *?DRINK*, then *?BEV* is an instance of Beverage

for all Cup *?CUP* holds if contains(*?CUP*, *?STUFF*), then *?STUFF* is an instance of Beverage

for all Tavern *?COMPANY* holds there exist CommercialService *?SERVICE*, beverage *?BEVERAGE* so that *?SERVICE* is an agent of *?COMPANY* and *?BEVERAGE* is a patient of *?SERVICE*

c) Utilisation dans le cadre de nos travaux

Nous avons utilisé SUMO pour identifier, dans WordNet, les sens d'un nom relatifs à un domaine donné. Par exemple, on obtient deux significations de CAT 'chat' en tant que « félin ». Cette possibilité de regrouper les sens de noms par domaine permet de se servir de WordNet avec un découpage des sens aussi bien fin (*fine-grained definitions*) que grossier (*coarse-grained definition*), cette dernière possibilité étant de nature à simplifier la désambiguïsation lexicale.

C. Extension de ces ressources

Nous présentons dans ce chapitre plusieurs expériences complémentaires que nous avons menées pour étendre le lexique sémantique à partir de WordNet et de la Wikipédia.

La section 2 présente l'appariement des synsets du Princeton WordNet avec des articles encyclopédiques (en l'occurrence, la Wikipédia). En capitalisant sur cette expérience, la section 3 (page 68) montre comment on peut extraire automatiquement des paraphrases à partir d'un corpus d'articles encyclopédiques comparables ; l'étape intermédiaire consiste alors à apparier un synset donné et les articles correspondants issus de plusieurs encyclopédies.

La section 4 (page 71) est largement indépendante des autres ; elle présente une méthode pour extraire automatiquement de WordNet des relations de polysémie régulière (comme la relation

entre une pièce de vaisselle et la quantité qui y est contenue), ainsi que les paires de lexies liées par de telles relations (CUILLER#1/CUILLER#2 par exemple). L'intérêt de cette ressource est notamment de fournir des informations dans certains contextes de désambiguïsation lexicale.

Enfin, la section 5 (page 79) propose une discussion sur la granularité des sens dans WordNet. Elle propose de regrouper les sens en sens macroscopiques pour faciliter la désambiguïsation lexicale.

1. Enrichissement du lexique par l'utilisateur

Antelope permet à l'utilisateur d'enrichir le lexique sémantique de base (correspondant aux données du Princeton WordNet) en créant des lexiques spécialisés. Ce mécanisme permet d'ajouter de nouveaux synsets, lemmes et relations, décrits dans un format XML[83]. Deux lexiques de ce type sont livrés avec Antelope, contenant :

- La traduction française de 44 200 lemmes, provenant de WOLF, le WordNet libre du français.
- 300 000 nouveaux synsets représentant des entités nommées (marques, produits, personnes, lieux…) correspondant à un sous-ensemble de la Wikipédia anglaise.

Nous allons à présent expliquer le mode opératoire ayant permis de construire ce second lexique.

2. Appariement de synsets de WordNet et d'articles encyclopédiques

(Ruiz-Casado, Alfonseca, Castells, 2005) présente l'implémentation d'un algorithme rapide permettant de réaliser la correspondance entre un article de la *Simple Wikipedia* et le synset correspondant de WordNet. Si aucun synset n'a de lemme en commun avec le titre de l'article, ce dernier est ignoré. Si un seul synset de WordNet a un lemme égal au titre, l'article y est lié sans autre analyse. En cas d'ambiguïté, l'article fait l'objet d'un étiquetage morphosyntaxique (après un filtrage des marqueurs syntaxiques spécifiques à la *Simple Wikipedia*), pour ne conserver que les noms, verbes et adjectifs. Le système analyse les définitions de WordNet et construit pour chacune d'entre elles un vecteur booléen (contenant « 1 » pour chaque terme en commun avec l'article et « 0 » pour chaque mot en disjonction). L'algorithme calcule alors une mesure de type cosinus entre les vecteurs, et retient le meilleur article, au sens de cette mesure de similarité. Les auteurs revendiquent une précision de 91,11 % (83,89 % sur les mots polysémiques).

Nous avons étendu et amélioré cet algorithme (Chaumartin, 2007b) avec une méthode permettant d'établir automatiquement une correspondance directe entre les articles d'une encyclopédie écrite en anglais (ici la *Simple Wikipedia* ou un sous-ensemble de l'*English Wikipedia*) et les entrées d'un lexique sémantique de référence (ici, les synsets de Princeton WordNet). Deux cas de figure se rencontrent alors : quand un article correspond déjà à une entrée du lexique, nous établissons la correspondance entre les deux ; sinon, nous enrichissons le lexique, en créant une nouvelle entrée et en la rattachant (via une relation d'hyperonymie/hyponymie) au meilleur « ancêtre » existant.

[83] C'est actuellement un format XML propriétaire, qui correspond à un sous-ensemble de SKOS. Nous prévoyons de mettre à jour ce format dans le futur pour être compatible avec le format standard SKOS.

Antelope est utilisée ici, d'une part pour effectuer une analyse syntaxique de la première phrase d'un article, de façon à détecter son genre prochain, d'autre part pour calculer une distance entre définitions, de façon à proposer des appariements.

Pour l'appariement entre WordNet et un sous-ensemble de 15 800 articles de la Wikipédia anglaise, nous obtenons une précision de 92 %. En cas de création d'un nouveau synset, l'hyperonyme est correctement identifié dans 85 % des cas.

a) Recherche des synsets de WordNet candidats à l'appariement avec un article encyclopédique

L'*English Wikipedia* possède une vingtaine d'articles dont le titre contient (au moins partiellement) « *Abraham Lincoln* » :

- « *Abraham Lincoln » :* l'homme politique, 16ème Président des Etats-Unis.
- « *Abraham Lincoln assassination* » : l'assassinat de l'homme politique.
- « *Abraham Lincoln (Pullman car)* » : le plus ancien wagon de passagers des Etats-Unis.
- Sans oublier deux films biographiques, trois lieux géographiques, plusieurs écoles, deux vaisseaux militaires… également nommés en mémoire de l'homme politique.

Nous constatons donc qu'une similarité entre le titre d'un article et un lemme (ou groupe de mots) désignant un synset de WordNet ne suffit pas à déduire qu'ils traitent du même sujet.

Notre approche consiste à identifier le (ou les) synset(s) de WordNet auquel un article se rattache. Pour ce faire, nous commençons par extraire de WordNet les « synsets candidats » pouvant correspondre au titre d'un article donné. Pour les personnes, par exemple, chaque article possède un ou plusieurs titres normalisés (de la forme « Prénom Nom » ou « Nom, Prénom »). Il suffit de rechercher les synsets correspondants dans WordNet. Pour un nom commun, il est nécessaire de tenir compte d'éventuelles variantes morphologiques et de retrouver la forme de base du mot.

Nous appliquons alors un ensemble d'heuristiques[84] pour retenir le meilleur candidat. S'il n'en existe pas, nous cherchons le synset correspondant le mieux à l'objet du monde décrit dans l'article (parle-t-on d'une rivière, d'un président… ?) Ensuite, nous créons un nouveau synset, rattaché (en tant qu'hyponyme ou instance hyponyme) au synset du thème de l'article, c'est-à-dire à son genre prochain.

b) Heuristiques utilisées dans notre approche

Notre approche améliore celle présentée dans (Ruiz-Casado, Alfonseca, Castells, 2005), avec deux différences. D'une part, nous avons ajouté plusieurs heuristiques, afin d'augmenter la précision. D'autre part, nous appliquons ces heuristiques même dans le cas où un seul synset de WordNet a un lemme égal au titre de l'article. Comme nous l'avons vu, l'*English Wikipedia* ne contient pas moins de vingt articles sur « *Abraham Lincoln* » ; cette décision permet d'éviter des appariements erronés.

Les heuristiques utilisées sont indépendantes les unes des autres ; elles peuvent donc être appliquées dans n'importe quel ordre. Au départ, tous les synsets candidats partent avec un même

---

[84] (Carré *et al.*, 1991) définit (p. 48) une heuristique comme « *une règle qu'on a intérêt à utiliser en général, parce qu'on sait qu'elle conduit souvent à la solution, bien qu'on n'ait aucune certitude sur sa validité dans tous les cas* ».

indice de confiance, qui est modifié durant l'application des heuristiques. Après cette étape, les synsets candidats qui disposent d'un poids manifestement trop faible pour correspondre à l'article sont supprimés de la liste. Dans notre cas, nous avons déterminé expérimentalement un poids minimal de 0,6. Ensuite, on conserve les synsets dont l'indice de confiance vaut au moins 40 % de celui du synset le mieux classé. Ceci permet de supprimer les synsets non significatifs.

(1) Distance vectorielle sur les mots

Cette heuristique est identique à celle décrite dans (Ruiz-Casado, Alfonseca, Castells, 2005).

(2) Comparaisons des contextes (domaines implicites et noms propres)

Nous extrayons du texte les domaines (« biologie», « sport »…) éventuellement associés à chaque mot[85], ainsi que les noms propres. Nous comparons la liste d'éléments extraits de l'article avec celle de chaque synset candidat, également à l'aide d'une mesure vectorielle.

(3) Comparaison des domaines cités explicitement dans le texte

Cette heuristique recherche, dans une définition, des patrons de la forme « *en mathématiques* », « *utilisé en géologie* »… à l'aide d'expressions régulières. Si un patron de ce type est repéré, son domaine d'application est extrait (« mathématiques » ou « géologie » par exemple). Si le synset candidat (ou l'un de ses hyperonymes) appartient à ce domaine, son indice de confiance est augmenté.

(4) Comparaison des hyperonymes

Cette heuristique a pour but de déterminer l'hyperonyme du sujet de l'article en étudiant sa définition. En voici quelques exemples, où les hyperonymes sont soulignés :

- **Abraham Lincoln** : 16ème Président des Etats-Unis.
- **Australie** : un pays et le continent le plus petit.
- **chat** : mammifère félin ayant une épaisse fourrure douce et incapable de rugir.

Le ou les hyperonymes du sujet de l'article sont comparés aux hyperonymes des synsets candidats. S'ils sont suffisamment proches (au sens d'une mesure de similarité), l'indice de confiance est fortement augmenté. Cette heuristique est essentielle en termes d'amélioration de la précision de l'appariement ; c'est pourquoi nous la détaillons ici.

(a) Analyse syntaxique de la définition

Notre but est d'extraire l'hyperonyme d'une définition. Prenons l'exemple précédent du « chat » ; notre but est d'extraire MAMMAL 'mammifère' (ou mieux FELINE MAMMAL 'mammifère félin', si ce terme existe dans le lexique de référence)[86].

Nous effectuons pour cela une analyse syntaxique en profondeur de la définition en utilisant le Stanford Parser. Cet analyseur statistique fournit une sortie sous forme de dépendances syntaxiques, comme montré en figure 17.

[85] Dans cette étape, nous comptons les domaines associés à chaque sens possible d'un mot du contenu de l'article.

[86] Si l'hyperonyme est qualifié par un adjectif ou un complément de nom, l'algorithme teste l'existence d'un synset constitué par l'expression complète, de façon à être le plus précis possible.

Nous supposons que l'hyperonyme se situe dans la 1ère phrase de l'article, qui tient le plus souvent lieu de définition ; nous ne traitons donc que celle-ci. Comme une définition se résume souvent à un groupe nominal, il convient de la modifier pour la rendre « grammaticalement correcte ». Notre expérience montre que c'est indispensable dans le cas d'un analyseur basé sur des règles comme le Link Grammar Parser et souhaitable dans le cas d'un analyseur statistique tel que le Stanford Parser. La première passe consiste donc en un étiquetage morphosyntaxique de la définition ; ensuite, en fonction de la partie du discours (adjectif, nom, verbe, etc.) du premier mot, l'algorithme préfixe éventuellement la définition par « *c'est* » ou « *c'est un* ».

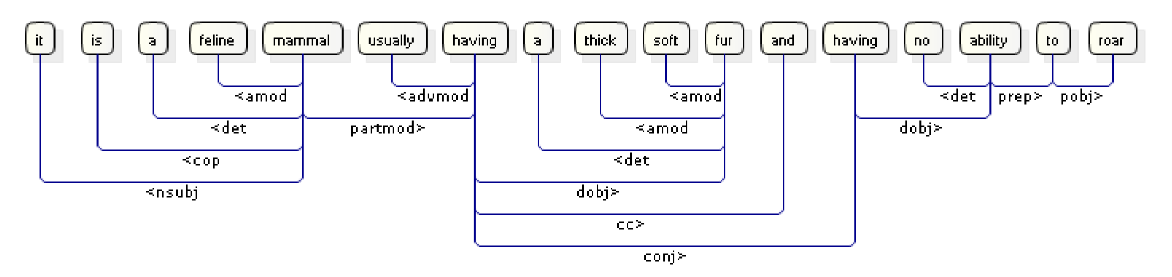


Figure 17 : Analyse syntaxique de la définition (en anglais) du nom « *chat* »

(b) Recherche de l'hyperonyme

L'analyse syntaxique de la définition est alors disponible sous forme d'un arbre de dépendances. Nous cherchons à y reconnaître le sous-arbre caractéristique d'une définition lexicographique en utilisant la méthode décrite dans (Chaumartin, 2006).

Le processus tient compte des conjonctions de coordination, afin d'extraire correctement les hyperonymes multiples comme dans « *l'Australie est un <u>pays</u> et le <u>continent</u> le plus petit* ». Dans une construction comme « *une espèce de…* » ou « *un membre du groupe de…* », nous remontons d'une façon récursive le long des constituants de l'amas nominal en passant au constituant imbriqué suivant.

(c) Création de nouveaux synsets

Si aucun synset de WordNet ne correspond à l'article considéré, on en crée un nouveau, dont la définition sera la première phrase de l'article. Ensuite on le relie au synset représentant l'hyperonyme de l'article étudié. On est confronté ici à une problématique de désambiguïsation lexicale, pour identifier le sens correct. Par exemple, si l'hyperonyme est « *empereur* », il faut choisir entre les sens « *dirigeant mâle d'un empire* », « *raisin rouge de Californie* » ou « *grand papillon richement coloré* ».

Les hyponymes du meilleur ancêtre se situent au même niveau que le sujet de l'article dans la hiérarchie de WordNet. Nous cherchons donc des points communs entre l'article et ses « cousins » potentiels. Nous commençons par relever les similarités au niveau du vocabulaire employé entre l'article et chacun des hyponymes de ses ancêtres possibles ; en effet, des articles ayant le même hyperonyme ont une forte probabilité de traiter de sujets voisins et de partager un champ lexical.

(5) Autres heuristiques

Pour finir, nous appliquons deux heuristiques supplémentaires. Tout hyperonyme candidat d'une entité nommée (personne, lieu, etc.) voit son indice de confiance augmenté s'il en découle des

relations de type « instance hyponyme », ou s'il hérite d'un groupe social (« *entreprise* », « *organisation* », « *mouvement* »…).

c) Résultats obtenus sur un sous-ensemble de la Wikipédia anglaise

La Wikipédia connait depuis plusieurs années une progression constante de son nombre d'articles[87] : certains ne sont que des biographies auto-promotionnelles, d'autres des comptes rendus de films ou de jeux vidéo… Notre choix a été de ne retenir ici que les entrées correspondant à un consensus en termes de connaissances encyclopédiques. Nous avons donc choisi de travailler sur un sous-ensemble des articles de l'*English Wikipedia* recoupant (sur la base du titre) ceux d'une autre encyclopédie de référence (la *Britannica online* en l'occurrence).

La version de mars 2006 de la Wikipédia en anglais (1 005 682 articles) a ainsi été filtrée pour retenir 15 847 articles qui ont été appariés automatiquement avec WordNet. Pour évaluer la précision de l'appariement, nous avons examiné manuellement le résultat sur 800 articles :

- 505 ont été associés à un synset existant déjà dans WordNet ; l'appariement a été fait correctement dans 465 cas (soit une précision de 92 %).
- 295 nouveaux synsets ont été créés ; l'hyperonyme a été correctement identifié dans 251 cas (soit une précision de 85 %).

Cette expérience montre qu'il est possible d'enrichir automatiquement WordNet à partir d'une encyclopédie. Un autre intérêt est d'augmenter la taille de la définition textuelle d'un synset ; cela permet en principe d'améliorer l'application de l'algorithme de Lesk de désambiguïsation lexicale (Cf. VII.B.5.c).

3. Apprentissage de paraphrases à partir de paires d'articles encyclopédiques comparables

a) Objectif

L'apprentissage automatique de paraphrases peut se faire sur la base de textes alignés ou comparables. (Ibrahim, Katz, Lin, 2003) décrivent ainsi l'utilisation de plusieurs traductions différentes, en anglais, d'œuvres littéraires (par exemple *20 000 lieues sous les mers*), et améliore l'approche de (Lin, Pantel, 2001) traitant de corpus comparables. L'algorithme mis en œuvre consiste à effectuer une analyse syntaxique de deux textes et à identifier le plus court chemin, dans chaque arbre de dépendance, entre deux ancres (typiquement des entités nommées).

Notre objectif est ici d'enrichir le lexique sémantique avec des cadres de sous-catégorisation associés à différentes constructions syntaxiques équivalentes, puis de constituer un catalogue de paraphrases dont les éléments sont totalement désambiguïsés par rapport au lexique sémantique. Pour cela, nous appliquons un algorithme proche sur des paires d'articles portant sur le même sujet.

[87] Par exemple, pour l'*English Wikipedia* : 3 650 000 articles en juin 2011 ; 1 540 000 fin 2006 ; 874 000 fin 2005 ; 414 000 fin 2004 ; 188 000 fin 2003 ; 95 000 fin 2002.

b) *Création de paires d'articles encyclopédiques comparables*

La méthode présentée en section IV.C.2 (page 64) montre comment apparier des synsets de WordNet et des articles encyclopédiques, avec une application à un sous-ensemble de l'*English Wikipedia*. En réitérant ce processus sur plusieurs encyclopédies, nous regroupons autour d'un synset donné plusieurs articles traitant d'un même sujet ; nous constituons donc ainsi un corpus monolingue d'articles comparables, propice à la découverte de paraphrases.

La figure 18 montre les articles de trois encyclopédies en langue anglaise[88] portant sur la rivière Alabama ; les entités nommées identiques sont surlignées dans une même couleur ; un module de résolution d'anaphores a été préalablement appliqué.

Wikipedia	Encyclopédie 2	Encyclopédie 3
The Alabama River, in the U.S. state of Alabama, is formed by the Tallapoosa and Coosa rivers, which unite six miles above Montgomery. The Alabama River flows west as far as Selma, then southwest until, about 45 miles from Mobile. The Alabama River unites with the Tombigbee to form the Mobile and Tensas rivers, which discharge into Mobile Bay.	The Alabama River is formed by the Coosa and Tallapoosa rivers northeast of Montgomery. The Alabama River winds westward to Selma and then flows south for a length of 318 mi. The Alabama River is joined above Mobile by the Tombigbee to form the Tensaw and Mobile rivers, which flow into the Gulf of Mexico.	The Alabama River is a river, 315 mi long, formed in central Alaska by the confluence of the Coosa and Tallapoosa rivers north of Montgomery. Flowing southwest to Mobile, Alaska, the Alabama River joins the Tombigbee to form the Mobile River.

Figure 18 : Comparaison de trois articles encyclopédiques anglais portant sur la rivière Alabama

Survolons le fonctionnement de l'algorithme, qui permet sur cet exemple de calculer que « *la rivière Alabama serpente jusqu'à Selma* » est une paraphrase de « *la rivière Alabama coule vers Selma* ». Nous représentons les paraphrases sous forme de triplets (sujet, verbe, complément). La désambiguïsation des entités nommées permet d'établir que « RIVIERE#1 serpente (préposition) VILLE#1 » est une paraphrase de « RIVIERE#1 coule (préposition) VILLE#1 »[89]. L'utilisation d'une mesure de similarité entre les deux verbes permet enfin de déterminer les sens précis des vocables SERPENTER et COULER dans le contexte. Nous obtenons, au final, l'équivalence entre deux cadres de sous-catégorisation[90], dont les éléments sont désambiguïsés par rapport au lexique : SERPENTER#1 (RIVIERE#1, VILLE#1) ~ COULER#2 (RIVIERE#1, VILLE#1).

c) *Traitement unitaire préalable d'un article*

Notre algorithme commence par traiter chaque article séparément, avec les étapes suivantes :

- Analyse syntaxique profonde du texte. Nous obtenons un ensemble de dépendances où les constructions de syntaxe de surface (sujet inversé…) sont gommées.

---

[88] Wikipédia en anglais ; Britannica online ; Columbia Electronic Encyclopedia.

[89] Rappelons que le suffixe #i indique le i^ème sens du mot dans le lexique.

[90] Nous avons conscience que, dans cet exemple, la présence d'un argument de type VILLE#1 dans le cadre de sous-catégorisation est discutable.

- Résolution des anaphores pronominales.
- Identification des entités nommées, autres que le sujet de l'article (donc autres que RIVIERE ALABAMA, dans notre exemple), et citées une seule fois (sans reprise anaphorique). Pour chacune de ces entités nommées :
    - Désambiguïsation lexicale (par rapport à WordNet).
    - Au sein d'une phrase donnée, recherche du plus court chemin reliant cette entité au sujet de l'article, dans le graphe de syntaxe profonde.

Précisons sur ce dernier point une limite importante de notre système actuel. Une paraphrase ne se limite généralement pas au remplacement d'un mot par un autre, mais plutôt d'un groupe de mots par un autre, sans que les deux groupes aient nécessairement la même taille. Or, pour des raisons de complexité calculatoire, nous nous sommes restreint dans cette expérience aux triplets de la forme (sujet, verbe, complément). Nous ne sommes donc actuellement en mesure de traiter que des cas simples de paraphrases. Lever cette contrainte permettra dans le futur de traiter des cas plus généraux en appariant des arbres de dépendances plus complexes.

En partant de l'article de l'*English Wikipedia* sur la rivière Alabama, nous obtenons ainsi des triplets de la forme (sujet, verbe, complément), dont le sujet et le complément sont déjà désambiguïsés : (RIVIERE COOSA, former, RIVIERE ALABAMA), (RIVIERE TALLAPOOSA, former, RIVIERE ALABAMA), (RIVIERE ALABAMA, couler, VILLE SELMA), (RIVIERE ALABAMA, unir, RIVIERE TOMBIGBEE), (RIVIERE ALABAMA, former, RIVIERE MOBILE)…

De même, un article d'une autre encyclopédie, traitant également de la rivière Alabama, fournit : (RIVIERE TALLAPOOSA, former, RIVIERE ALABAMA), (RIVIERE COOSA, former, RIVIERE ALABAMA), (RIVIERE ALABAMA, serpenter, VILLE SELMA), (RIVIERE TOMBIGBEE, rejoindre, RIVIERE ALABAMA), (RIVIERE ALABAMA, former, RIVIERE MOBILE)…

d) Rapprochement des informations entre paires d'articles

Nous pouvons alors rapprocher ces informations. En enlevant les triplets identiques, il reste (RIVIERE ALABAMA, couler, VILLE SELMA) ~ (RIVIERE ALABAMA, serpenter, VILLE SELMA) et (RIVIERE ALABAMA, unir, RIVIERE TOMBIGBEE) ~ (RIVIERE TOMBIGBEE, rejoindre, RIVIERE ALABAMA). Les entités nommées sont déjà désambiguïsées ; connaissant leurs hyperonymes, nous pouvons donc réécrire ces paraphrases au niveau des classes plutôt que des instances :

- (RIVIERE#1 riv1, couler, VILLE#1 v1) ~ (RIVIERE#1 riv1, serpenter, VILLE#1 v1)
- (RIVIERE#1 riv1, unir, RIVIERE#1 riv2) ~ (RIVIERE#1 riv2, rejoindre, RIVIERE#1 riv1)

e) Application d'une mesure de similarité aux verbes des paraphrases

Il nous reste à déterminer le sens de chacun des deux verbes dans la paire de triplets. Nous utilisons pour cela la mesure de similarité « structurelle » (présentée page 80), qui exploite la hiérarchie du graphe d'hyperonymes des verbes de WordNet. Partant de l'hypothèse que les deux verbes doivent avoir un sens proche l'un de l'autre, nous cherchons la combinaison de sens qui minimise leur distance, au sens d'une telle mesure.

Nous appliquons cette mesure de similarité à toutes les combinaisons de sens de « couler » et « serpenter », d'une part, et d' « unir » et « rejoindre », d'autre part. Nous obtenons alors, comme combinaison minimisant la distance entre les paires de verbes :

- (RIVIERE#1 riv1, COULER#2, VILLE#1 v1) ~ (RIVIERE#1 riv1, SERPENTER#1, VILLE#1 v1)
- (RIVIERE#1 riv1, UNIR#4, RIVIERE#1 riv2) ~ (RIVIERE#1 riv2, REJOINDRE#5, RIVIERE#1 riv1)

f) Bilan

Dans l'optique d'une validation semi-automatique des paraphrases proposées, on peut envisager de procéder au traitement de l'ensemble des articles d'une catégorie donnée (par exemple, tous les articles décrivant des rivières). Cette restriction permet de rester à l'intérieur d'un champ thématique et augmente les chances de trouver plusieurs occurrences de la même paraphrase. On peut alors compter la fréquence de chaque paraphrase et fixer un seuil minimal en dessous duquel elle n'est pas retenue ; cette approche permet en principe de compenser les erreurs ayant pu subvenir dans la chaîne de traitement, durant les phases d'analyse syntaxique, de désambiguïsation lexicale des entités nommées ou de résolution d'anaphores. Si une même paraphrase se retrouve plusieurs fois, elle est probablement correcte.

Cette expérience montre qu'il est possible, en disposant de plusieurs textes portant sur un même sujet, d'extraire automatiquement des paraphrases, avec des cadres de sous-catégorisation dont les constituants sont désambiguïsés par rapport à WordNet. Nos évaluations préliminaires (effectuées sur une dizaine d'articles) montrent une précision de l'ordre de 70 % dans la détection de paraphrases pertinentes. Il reste à mettre en œuvre ces mécanismes sur un volume significatif d'articles pour affiner notre jugement sur la validité de cette approche.

Ces cadres de sous-catégorisation fournissent de puissants indices de désambiguïsation lexicale, qui peuvent être utilisés lors de traitements ultérieurs.

4. Extraction de relations de polysémie régulière

Nous avons procédé (Barque, Chaumartin, 2008 ; Barque, Chaumartin, 2009) à une analyse et une modélisation des relations de polysémie régulière. Cette étude exploite la hiérarchie des noms et verbes de WordNet et la définition associée à chacun de ces synsets. Un ensemble de règles a permis d'identifier d'une façon largement automatisée 2 350 instances de relations de métaphore et de métonymie, avec une précision voisine de 91 %. La méthode utilisée permet aussi d'obtenir une désambiguïsation lexicale partielle de la définition associée aux synsets.

Nous commencerons par dresser un rapide état des lieux des recherches dédiées à la description de la polysémie régulière. Ensuite, nous exposerons les objectifs de cette étude et les moyens mis en œuvre pour y parvenir ; nous proposerons notamment une méthode de construction des patrons de polysémie « assistée par ordinateur ». Enfin, nous présenterons les résultats obtenus, sous forme d'une classification de ces patrons et d'une mesure de leur régularité dans WordNet.

a) Etat de l'art

WordNet a déjà été exploité en vue d'une caractérisation de la polysémie (Peters, 2006 ; Veale, 2006). Dans cette lignée, notre expérience propose une démarche pour créer des patrons de polysémie régulière, puis pour détecter automatiquement leurs occurrences dans ce lexique. Considérons la définition lexicographique de l'un des synsets de WordNet :

- {TREACHERY#2, **BETRAYAL#1**, TREASON#3, PERFIDY#2} (*an act of deliberate **betrayal***)

On remarque dans cet exemple qu'un des éléments du synset se retrouve dans sa définition (BETRAYAL#1 est partiellement défini avec le mot *betrayal*). Plus précisément, la lexie BETRAYAL#1 contient dans sa définition une autre[91] lexie du vocable BETRAYAL ; toutefois, on ignore laquelle avant d'avoir désambiguïsé les éléments de la définition.

(1) Description de la polysémie régulière

L'intérêt d'expliciter la polysémie régulière lors du développement d'un lexique a souvent été mis en évidence, notamment dans le cadre du TAL. Qu'elle soit représentée sous forme de règles lexicales (Ostler, Atkins, 1991; Copestake, Briscoe, 1995) ou de mécanismes transformationnels agissant lors de la composition de mots en syntagmes (Pustejovsky, 1995), la description de la polysémie régulière présente au moins deux avantages.

D'un point de vue théorique tout d'abord, il s'agit d'offrir une représentation de l'un des aspects de la formation du lexique, la polysémie régulière constituant une source importante de créativité lexicale. Par exemple, en disposant d'une règle de polysémie régulière entre une unité de type {ANIMAL} et une unité de type {PERSONNE}[92], le lexique dispose potentiellement d'entrées de type {PERSONNE} dérivées à partir d'entrée existantes de type {ANIMAL}. Cette alternance s'applique par exemple aux vocables GORILLE, LIEVRE, TAUPE, REQUIN, etc. En revanche, l'unité de type {PERSONNE} pour le vocable MULE (*individu chargé de transporter de la drogue*) n'est pas (encore) référencée dans les dictionnaires, même s'il apparaît de plus en plus souvent dans les textes journalistiques, parfois entre guillemets[93]. Autrement dit, les règles de polysémie constituent l'un des moyens de rendre compte de l'aspect dynamique du lexique, ce qui est intéressant dans certains algorithmes de désambiguïsation.

Le second avantage, d'ordre pratique, concerne la valorisation du lexique à partir duquel s'effectue l'étude de la polysémie régulière, en l'occurrence WordNet. Les règles lexicales de polysémie régulière permettent en effet de systématiser l'encodage des données en fournissant au lexicographe un canevas définitionnel. Par exemple, le schéma de définition **L2 = quantité de X contenue dans L1** pourra servir à la définition d'autres paires de lexies de type `pièce de vaisselle` ~ `quantité (de qqch.)` liées par une métonymie régulière :

- ASSIETTE#2 de X = quantité de X contenue dans une ASSIETTE#1
- BOL#2 de X = quantité de X contenue dans un BOL#1

(2) Définition de la polysémie régulière

Selon Apresjan (1974), une polysémie est régulière s'il existe au moins deux vocables (A et B) ayant chacun deux lexies (A#1~A#2 et B#1~B#2) liées par la même relation sémantique. Les lexies A#1 et B#1 ne doivent pas être synonymes, pas plus que les lexies A#2 et B#2. Illustrons cette notion de polysémie régulière avec des données extraites de WordNet :

---

[91] Notre hypothèse est qu'il s'agit forcément d'une autre lexie, sinon la définition serait récursive.

[92] Idéalement, WordNet aurait pu définir un synset {PERSONNE AYANT UNE FONCTION} hyponyme de {PERSONNE}.

[93] « *La **mule** avait ingéré 90 boulettes en plastique contenant la drogue* » (Romandie, 20/10/2011). « *Il devrait mettre un coup de frein à une faille du Code de procédure des douanes sur la remise en liberté des **mules*** » (DNA, 15/10/2011). « *Il s'appelle David et c'est une « **mule** », un passeur de cocaïne* » (Le Figaro, 21/10/2011).

- {CERISE#1, **CHERRY**#4, CHERRY RED#1} (*the red color of **cherries***)
- {**CHESTNUT**#4} (*the brown color of **chestnuts***)

Les vocables CHERRY et CHESTNUT présentent la même alternance sémantique entre un {FRUIT} et une {COULEUR}, respectivement le rouge des cerises et le marron des châtaignes. On peut donc d'ores et déjà dire que ce lien est régulier et chercher d'autres occurrences dans WordNet, afin de déterminer son degré de régularité (Barque, 2008). Bien entendu, ce dernier dépendra du degré de spécificité de la caractérisation sémantique du lien. Par exemple, le lien entre {FRUIT} et {COULEUR} sera associé à moins d'occurrences que le lien entre une {ENTITE} et une {COULEUR}, le type {ENTITE} étant plus général que le type {FRUIT}.

Une chose est de déterminer le degré de régularité d'un lien de polysémie, une autre de déterminer à quelle catégorie il appartient. Nous distinguons dans la suite trois grandes catégories de liens de polysémie régulière (Fass, 1988).

- Une lexie L2 est une **spécialisation** d'une lexie L1 si son sens est plus spécifique que celui de L1. Ci-dessous, on peut voir que la lexie PRESSURE#7 dénote un cas particulier de ce à quoi renvoie *"pressure"* dans la définition : {**PRESSURE**#7} (*the **pressure** exerted by the atmosphere*).
- Deux lexies L1 et L2 sont liées par **métaphore** si leurs deux référents sont en relation d'*analogie*, autrement dit s'ils se ressemblent sur au moins un de leurs aspects. Par exemple, le rire dénoté par la lexie CACKLE#3 ressemble, du point de vue du son, au gloussement de la poule (*"hen's cackle"*) : {**CACKLE**#3} (*a loud laugh suggestive of a hen's **cackle***).
- Deux lexies L1 et L2 sont liées par **métonymie** si le référent de L1 et celui de L2 sont en relation de *contiguïté*, autrement dit si les deux référents « se touchent », de façon plus ou moins concrète. Par exemple, le lien entre les deux sens de CHESNUT, déjà évoqué plus haut, relève de la métonymie puisque la couleur dénotée par la lexie chesnut#4 est celle du fruit dénoté par *"chesnuts"* dans la définition : {**CHESNUT**#4} (*the brown color of **chestnuts***).

b) Objectif et méthode

L'objectif de notre expérience est d'enrichir WordNet pour de nouvelles applications. Pour cela, nous allons décrire les liens de polysémie réguliers de WordNet et mesurer leur régularité en détectant automatiquement leurs occurrences dans la base lexicale. L'un des enrichissements possibles du lexique sera ainsi la création de nouvelles relations sémantiques, en l'occurrence des relations de métaphore et de métonymie[94].

(1) Description des règles

Notre objectif étant de rendre compte de la polysémie **régulière** représentée dans WordNet, nous avons pris le parti de nous intéresser, dans un premier temps, aux seuls synsets dont la définition contient une lexie (L1) appartenant au même vocable que l'une des lexies du synset défini (L2) ; ce parti pris repose sur l'idée communément admise qu'un lien de sens entre deux lexies de même forme est d'autant plus évident que l'une est définie au moyen de l'autre : {…, **L2**, …} = … ***L1*** … En voici deux exemples :

- {**DRIVER**#3} (*a golfer who hits the golf ball with a **driver***)
- {**FALSIFY**#4} (***falsify** knowingly*)

[94] Dans WordNet, les relations de spécialisation sont déjà présentes pour les noms et verbes. Par exemple, PRESSURE#7 a explicitement pour hyperonyme PRESSURE#1.

Dans le premier exemple, la lexie **DRIVER**#3 est définie au moyen d'une autre lexie du vocable DRIVER. Rappelons qu'à ce stade, cette dernière n'est pas identifiée, les éléments utilisés dans les définitions de WordNet n'étant pas désambiguïsés[95]. Sur le plan informatique, nous avons procédé à l'étiquetage morphosyntaxique des définitions de tous les synsets pour les filtrer et retenir un premier ensemble de 1984 synsets où L1 et L2 appartiennent à la même partie du discours. En toute rigueur, nous avons imposé des contraintes supplémentaires ; nous avons éliminé les synsets où L1 désigne en fait L2 ; cela correspond aux cas où la définition contient :

- *"equal*" comme dans {KOPEK, KOPECK, COPECK} (*100 kopecks equal 1 ruble in Russia*).
- *"trademark*" ou *"trade name*" car L1 et L2 représentent alors un nom commercial, comme dans {SILDENAFIL, SILDENAFIL CITRATE, VIAGRA} (*virility drug, trade name Viagra*)).
- "*capital of*" comme dans {BERN, BERNE, CAPITAL OF SWITZERLAND} (*the capital of Switzerland*).

La méthode adoptée pour attribuer une catégorie de liens de polysémie à une occurrence L1~L2 repose sur différents critères formels appliqués aux définitions de WordNet (Martin, 1972 ; Fass, 1988). Nous avons regardé, tout d'abord, si l'inclusion de L1 figurait dans la première partie de la définition de L2 (*i.e.* en tant que genre prochain) ou bien dans sa seconde partie (*i.e.* en tant que différence spécifique), comme illustré ci-dessous :

- {**BEHAVE**#3} (***behave** well or properly*)
- {**SWEEP**#6} (*clean by **sweeping***)

Dans le premier exemple, L1 apparaît dans la première partie de la définition de L2 : le troisième sens du verbe BEHAVE est défini au moyen d'un autre sens du même vocable, qui en constitue le genre prochain (« *se comporter* » signifie, dans un de ses sens, « *se comporter d'une certaine manière »,* en l'occurrence d'une manière appropriée). Dans le second exemple, L1 apparaît dans la seconde partie de la définition de L2 : le sixième sens du verbe TO SWEEP ne veut pas dire « *balayer d'une certaine manière »* mais « *faire quelque chose en balayant* », en l'occurrence nettoyer en balayant.

Outre la place de l'inclusion de L1 dans la définition de L2, nous retenons des sous-chaînes récurrentes dans les définitions. Voici trois exemples d'inclusion dans la seconde partie de la définition, distingués selon certains éléments pertinents de leur définition :

- {**MINT**#5} (*a candy that is flavored with a **mint** oil*) : "that is flavored with L1"
- {**BLUEFISH**#2} (*fatty bluish flesh of **bluefish***) : "flesh of L1"
- {**FIN**#5} (*a stabilizer on a ship that resembles the **fin** of a fish*) : "that resembles L1"

On peut ainsi, en mêlant ces deux critères (la place de l'inclusion et les éléments définitionnels qui entourent cette inclusion), attribuer de manière automatique une catégorie de lien de polysémie à une occurrence donnée. De manière informelle, disons que si l'inclusion a lieu dans la première partie de la définition, il s'agit soit d'une spécialisation, soit d'une métaphore. Les deux exemples ci-dessous montrent en effet deux cas d'inclusion de L1 dans la première partie de la définition ; mais le premier exemple relève de la spécialisation, tandis que le second relève de la métaphore.

[95] Voir néanmoins la section sur les gloses désambiguïsées (page 53). A la date où nous avions réalisé cette expérience, nous ne disposions que d'Extended WordNet. Le ratio de validation humaine de désambiguïsation des gloses étant faible (2,3%), nous n'avions pas utilisé cette ressource.

- {**ARRANGE**#5} (***arrange** attractively*)
- {**GROW**#9} (***grow** emotionally*)

Si l'inclusion a lieu dans la seconde partie de la définition, il s'agit soit d'une métonymie, soit d'une métaphore, selon le type d'élément qui introduit l'inclusion. Parmi les trois exemples déjà présentés plus haut, les deux premiers, {**MINT**#5} et {**BLUEFISH**#2}, sont des cas de métonymie, le troisième ({**FIN**#5}) est un cas de métaphore. Ici, l'ambiguïté sur la catégorie d'appartenance d'un lien donné peut être levée grâce aux canevas définitionnels. Par exemple, si l'inclusion est précédée de la séquence *"that resembles"* comme c'est le cas pour le vocable FIN 'nageoire' ci-dessus, on sait qu'il s'agit d'un cas de métaphore et non de métonymie.

(2) Recherche d'occurrences

Notre méthode de détection des liens de polysémie régulière s'applique, dans un premier temps, à l'ensemble des 1984 synsets dont la définition inclut un synset de même forme. Nous avons créé manuellement une soixantaine de patrons en analysant des définitions de ces synsets qui correspondaient manifestement à un cas de polysémie régulière.

L'application de ces patrons a permis d'obtenir un premier classement de 1427 synsets[96]. Nous allons détailler ce processus en montrant notamment comment il permet de désambiguïser L1. Nous verrons ensuite comment cette méthode peut se généraliser aux autres synsets ne présentant pas la particularité d'inclure un synset de même forme dans leur définition.

(a) Définition de patrons de polysémie régulière

Voici quelques lignes de code[97] définissant un patron appelé `colorOf` (« couleur de ») :

```
patterns.Add(new Pattern("colorOf")
   .AddType("color", "fruit")
   .AddType("color", "gem")
   .AddType("color", "metal")
   .AddMatchingRule("color of *"));
```

La première ligne de code définit le patron de polysémie. Les trois lignes suivantes indiquent que les paires de lexies susceptibles d'instancier ce patron sont de type {COLOR} pour L2 et {FRUIT}, {GEM} ou {METAL} pour L1. Enfin, la dernière ligne indique que la définition de L2 doit, pour être déclarée occurrence du patron, contenir la chaîne de caractères *"color of"* suivie de L1 (indicateur `*` à droite)[98].

Considérons un autre patron de polysémie produit par notre étude :

```
patterns.Add(new Pattern("causedBy")
   .AddMatchingRule("resulting from *")
   .AddMatchingRule("caused by *"));
```

Contrairement au précédent, ce patron n'impose pas de contrainte sur le type des lexies susceptibles de l'instancier ; en revanche, L2 doit alors inclure dans sa définition une des deux séquences

---

[96] Les autres 557 synsets correspondent soit à des situations n'étant pas un cas de polysémie régulière, soit à des situations où la définition d'un patron de polysémie ne permettrait de couvrir qu'un faible nombre de cas.
[97] Cet exemple est codé en langage C#.
[98] Plusieurs paraphrases peuvent être précisées ; par exemple, la détection des métaphores testera plusieurs cas de figure : *"suggestive of *", "similar to *", "corresponds to *", "that suggests *", "imitating *"*…

indiquées (“*resulting from*” ou “*caused by*”). On voit ici que la seule utilisation de la hiérarchie des concepts conduirait à exclure un certain nombre de patrons de polysémie régulière et donc à diminuer le rappel de notre méthode.

(b) Application des patrons

Les exemples ci-après sont reconnus comme des occurrences du patron `colorOf` :

- {**EMERALD**#3} (*the green color of an **emerald***)
- {**TAN**#2, **TOPAZ**#3} (*a light brown, the color of **topaz***)
- {**COPPER**#4} (*a reddish-brown color resembling the color of polished **copper***)

Grâce aux informations de typage associées au patron, la lexie L1 peut être ensuite désambiguïsée. Pour ce faire, le système énumère tous les sens possible de L1 (autres que L2), et s’arrête quand le couple (L2, L1) est compatible avec l’un des couples de types définis dans le patron (la hiérarchie d’hyperonymie des noms ou des verbes est explorée si besoin).

On obtient alors une définition dans laquelle le sens de L1 est désambiguïsé :

- {**EMERALD**#3} (*the green color of an **EMERALD**#1 $_{[gem]}$*)
- {**TAN**#2, **TOPAZ**#3} (*a light brown, the color of **TOPAZ**#2 $_{[gem]}$*)
- {**COPPER**#4} (*a reddish-brown color resembling the color of polished **COPPER**#1 $_{[metal]}$*)

(c) Généralisation par élargissement du champ d’application des patrons

Les patrons sont ensuite appliqués à l’ensemble des lexies ayant plusieurs sens, sans imposer aux synsets la contrainte d’inclure un synset de même forme dans leur définition. Cette étape permet d’identifier 367 synsets supplémentaires. Par exemple, GOLD ‘or’ a cinq sens : les pièces d’or, la couleur, le métal, une bonne santé, quelque chose de précieux. La paire de lexies (GOLD#2, GOLD#3) est de type ({COLOR}, {METAL}) ; elle est donc compatible avec la règle `colorOf`, et peut donc s’appliquer même si la définition de {GOLD#2} (*a deep yellow color*) ne contient pas directement le vocable GOLD. Le même traitement est appliqué pour {CORAL#1}. On obtient alors aussi :

- {AMBER#1, GOLD#2} (*a deep yellow color*) : lien implicite vers GOLD#3$_{[metal]}$
- {CORAL#1} (*a variable color averaging a deep pink*) : lien implicite vers CORAL#2$_{[gem]}$ ‘corail’

Cette généralisation de l’application des patrons doit toutefois se faire en prenant des précautions. On constate expérimentalement que cette généralisation donne de bons résultats sur certains patrons, mais pas sur tous. En effet, quand les patrons sont contraints par des types trop généraux (entité, artefact, abstraction…), le fait de ne plus imposer L1 dans la définition de L2 va se traduire par une multiplication de couples de synsets qui ne sont pas liés par une relation de polysémie régulière. Pour minimiser ce risque, un patron peut tester si les deux synsets portent effectivement sur le même sujet ou des sujets voisins. Cette vérification est implémentée par une classique mesure de similarité entre les deux définitions, qui peut être astreinte à respecter un seuil minimal. Nous avons utilisé une mesure vectorielle de recouvrement des mots entre définitions avec une pondération de type TF-IDF. Par exemple, parmi les occurrences de la relation entre un mouvement et le son associé, nous obtenons pour « *(bruit de) pas* » une paire de synsets dont les définitions comportent deux mots en commun (donnant une similarité égale à 48,5 %) :

- {FOOTSTEP#1} (*the sound of a **step** of someone **walking***)
- {FOOTSTEP#2} (*the act of taking a **step** in **walking***)

Imposer une telle contrainte, avec un seuil minimal, favorise la précision au détriment du rappel. Par exemple, pour la métaphore entre animal et personne, notre système identifie « *tigre* » (et il existe bien une métaphore entre les deux lexies, basée sur la férocité de l'animal) ; néanmoins, il ne le retient pas car les deux définitions ne partagent aucun mot :

- {TIGER#1} (*a fierce or audacious person*)
- {TIGER#2} (*large feline of forests in Asia having a tawny coat with black stripes*)

c) Résultats

Nous proposons ici une classification (non exhaustive) des relations de métonymie et de métaphore[99] sur la base des patrons identifiés pendant l'étude. Dans cette classification, nous indiquons entre parenthèses deux nombres (occurrences correctes / nombre total d'occurrences détectées), suivis d'exemples significatifs choisis pour illustrer le caractère régulier de la relation.

(1) Classification des relations de métonymie

L2 *représente* L1
└─ Carte à jouer *représente* Figure ou Nombre entier (5/6 ; QUEEN#7, KING#9 ; TEN#2, NINE#3)
L2 *est causé par* L1
├─ Dépense *causée par* Action (27/27 ; ADMISSION#3, ANCHORAGE#2)
└─ Maladie *causée par* Organisme (13/17 ; ERGOT#1, HERPES#1)
L2 *est produit par* L1
├─ Son *produit par* Instrument, Mouvement ou Appareil (15/15 ; DRUM#2, WHISTLE#1 ; BELL#3)
├─ Œuvre *écrite par* Personne
│ ├─ Livre *écrit par* Auteur (pas d'exemple dans WordNet ; ce pourrait être "Shakespeare")
│ │ └─ Livre *écrit par* Prophète (15/15 ; JOB#12, JEREMIAH#2)
│ └─ Musique *écrite par* Compositeur (9/9 ; MOZART#2, WAGNER#3)
└─ L2 *produit par* Plante ou Arbre
  ├─ Fruit *produit par* Arbre (ORANGE#1, CITRUS#1)
  └─ Fleur *produite par* Plante (50/51 ; CHRYSANTHEMUM#1, COTTONWEED#1)
L2 *produit* L1
└─ Entreprise *produit* Media (2/2 ; NEWSPAPER#2, MAGAZINE#3)
L2 *est dérivé de* L1
├─ L2 *est dérivé d'*Animal
│ ├─ Chair *d'*Animal, Poisson, Volaille ou Crustacé (303/303 ; RABBIT#3 ; TROUT#1 ; PHEASANT#2)
│ ├─ Fourrure *d'*Animal (17/17 ; FOX#3, CHINCHILLA#1)
│ └─ Laine *d'*Animal (2/2 ; ALPACA#1, VICUNA#1)
└─ L2 *est dérivé de* Plante, Feuille, Arbre…
  ├─ Boisson *dérivée de* Feuille (3/3 ; TEA#1, MATE#9)
  ├─ Fibre *dérivée de* Plante (13/13 ; COTTON#1, FLAX#1)
  ├─ Bois *dérivé d'*Arbre (70/70 ; BAMBOO#1, BALSA#1)
  └─ Vin *dérivé de* Vigne (2/2 ; TOKAY#1, VERDICCHIO#2)
L2 *a pour sujet* L1
├─ Discipline *a pour* Sujet (56/64 ; LITERATURE#2, PHYSICS#1)
├─ L2 *est responsable de* L1 (4/6)
│ ├─ Ministère *est responsable de* Sujet (EDUCATION#6, ENERGY#7)
│ └─ Division *est responsable de* Sujet (PERSONNEL#2, SECURITY#6)
└─ Livre *a pour sujet* Personne (6/6 ; JONAH#3, JOSHUA#2)
L2 *accompagne* L1
└─ Musique *accompagne* Danse (32/32 ; POLKA#1, MAZURKA#1)
L2 *est inclus dans* L1

[99] Les résultats contiennent également une proportion significative d'occurrences de liens de spécialisation (approximativement 12%). Toutefois, il nous semble que cette catégorie de lien de polysémie se prête mal à une classification, dans la mesure où il est difficile d'identifier un typage régulier pour L1 et L2.

├─ Substance *contenue dans* Médicament (17/17 ; ARNICA#2, MENTHOL#1)
├─ Personne *membre de* Groupe (37/39 ; SAMURAI#1, NINJA#1)
├─ Personne *occupant une* Construction (6/6 ; BUILDING#4, FLOOR#7)
├─ Quantité *contenue dans* Conteneur (39/39 ; TEASPOON#1, BAG#5)
│ └─ Nourriture *contenue dans* Plat (5/5 ; PLATE#8, CASSEROLE#1)
├─ Rivière *passant dans* Région (6/6 ; ALABAMA#3, DELAWARE#1)
└─ Pays *situé dans* Île (22/22 ; IRELAND#1, MALTA#1)

L2 *est caractérisé par* L1
├─ Balle *caractérisé par* Jeu (9/10 ; PAINTBALL#1, VOLLEYBALL#2)
├─ Vin *provenant de* Région (4/4 ; CHABLIS#2, BORDEAUX#2)
├─ Couleur *caractéristique de* L1 (7/7)
│ ├─ Couleur *caractéristique de* Gemme (TOPAZ#3, EMERALD#3)
│ ├─ Couleur *caractéristique de* Métal (GOLD#2, COPPER#4)
│ └─ Couleur *caractéristique de* Fruit (CHERRY#4, CHESTNUT#4)
├─ Nourriture *au goût de* L1 (13/25)
│ └─ Nourriture *au goût d'*Herbe (MINT#5, RATAFIA#2)
├─ Vêtement *caractérisé par* Partie du corps (12/14 ; BACK#7, SHOULDER#4)
└─ Personne *caractérisée par* L1
  ├─ Sportif *caractérisé par* Position (31/31 ; CENTER#13, WINGBACK#1)
  └─ Chanteur *caractérisé par* Voix (11/11 ; CONTRALTO#1, SOPRANO#1)

Langue *parlée par* Personne (199/223 ; KOREAN#2, PORTUGUESE#1)

(2) Classification des relations de métaphore

L2 *est analogue à* L1
├─ Communication humaine *est analogue à* Communication animale (3/4 ; BARK#1, CACKLE#1)
├─ Partie du corps animal *correspond à* Partie du corps humain (3/3 ; LEG#2, THROAT#4)
├─ Individu *ressemble, dans son comportement, à* Animal (36/54 ; PIRANHA#1, POPINJAY#1 (*perroquet*))
└─ Objet *ressemble, par sa forme, à* Objet naturel (38/38 ; MOON#2, SNAKE#5)
  └─ Artefact *ressemble à* Partie du corps (5/5 ; NOSE#2, THROAT#3)

(3) Evaluation des résultats

À notre connaissance, il n'existe pas de standard de référence pour ce type d'expérience. Nous avons évalué manuellement les 2 351 relations proposées par notre système. Nous estimons que 2 140 d'entre elles sont correctes, ce qui donne une précision de 91,03 %.

Nous n'avons pas identifié de méthode permettant une évaluation automatique précise du rappel. Toutefois, nous avons calculé manuellement le rappel pour deux des patrons de polysémie présentés ci-dessus : la métaphore *Individu ressemble à Animal*, ainsi que la métonymie *Bois dérivé d'Arbre*. Nous avons identifié manuellement 142 occurrences du lien de métaphore dans WordNet (rappel de 36/142 = 25,3 %) et 79 occurrences du lien de métonymie (rappel de 70/79 = 88,6 %). Comme on le voit, le rappel dépend aussi de la nature de la relation, qui peut être plus ou moins régulière.

d) Application à la désambiguïsation lexicale

La ressource produite par cette méthode peut être mise à contribution dans une tâche de désambiguïsation lexicale, pour inférer des sens de mots qui n'existent pas explicitement dans le lexique. Nous allons voir comment elle permet de créer dynamiquement des nouveaux sens, quand le contexte s'y prête.

Pour illustrer comment nous remédions aux imperfections du lexique, choisissons l'exemple de métonymie *Vin provenant de Région*. Dans WordNet, BORDEAUX[100] et CHABLIS apparaissent avec les deux sens (vin et région). En revanche, BOURGOGNE[101] n'y figure qu'en tant que région. (Pour

[100] {BORDEAUX#1} (*a port city in southwestern France*) ; {BORDEAUX#2, BORDEAUX WINE#1} (*any of several red wines or white wines produced around Bordeaux*).
[101] {BOURGOGNE#1, BURGUNDY#1} (*a former province of eastern France that is famous for its wines*).

information, mais ce n'est pas essentiel ici, le seul sens présent pour CHIANTI[102], BEAUJOLAIS, MEDOC et RIOJA est celui du vin.)

Considérons la phrase "*Friends don't let friends drink Bourgogne*". Les deux premiers sens du verbe DRINK dans WordNet peuvent convenir[103] (boire / boire de l'alcool). Ils ont le même cadre de sous-catégorisation associé dans VerbNet ; sa construction « *Agent* boit *Patient* » précise que *Patient* doit avoir les traits <concret+comestible+liquide>. BOURGOGNE#1 est donc incompatible avec ces contraintes de sélection ; sans information complémentaire, notre mécanisme d'étiquetage de rôles thématiques utilisant VerbNet et WordNet échouera à trouver une solution.

La prise en compte de la métonymie permet de résoudre ce problème, en autorisant une opération de « coercition de type ». (Pustejovsky, 1995) postule que la multiplicité des sens des mots s'explique par des mécanismes génératifs universels. Chaque lexème possèderait un certain degré d'ambiguïté (polysémie logique) et des mécanismes généraux permettent la sélection du sens correct en contexte. Si on admet que le sens d'un mot peut « glisser » (par exemple, en cas de polysémie régulière), le mécanisme de coercition de type permet au verbe de convertir et de contraindre le type de ses arguments s'ils ne conviennent pas.

En cas d'impossibilité à trouver dans WordNet un sens respectant les contraintes de sélection pour l'un des arguments, le système peut tenter d'appliquer une opération de coercition de type sur cet argument. Dans notre exemple, le système sait (grâce à WordNet) que BOURGOGNE#1 est une région ; la connaissance des relations de polysémie régulière (dont la métonymie *Vin provenant de Région*) permet de proposer un sens virtuel (non matérialisé dans WordNet) BOURGOGNE#2 (*wine from the Bourgogne region*) hyponyme de WINE#1 'vin' ; ce sens dynamique est donc également hyponyme de BEVERAGE#1 'boisson' et satisfait aux contraintes de sélection <concret+comestible+liquide>.

e) Conclusion et perspectives

Nous avons présenté ici une méthode permettant de créer automatiquement dans (et à partir de) WordNet, avec une bonne précision, deux nouvelles catégories de relations sémantiques, métaphores et métonymies. La ressource contenant ces nouvelles relations est disponible en ligne[104].

Ce travail a été effectué sur l'anglais. Il pourrait aussi être décliné pour des WordNet en d'autres langues (une fois, bien sûr, les patrons adaptés à la langue décrite) pour (*i*) aider à valider l'homogénéité des définitions produites et (*ii*) comparer les polysémies régulières partagées entre différentes langues.

5. Granularité des sens dans le lexique

a) Mesures de similarité entre synsets

Les algorithmes effectuant des traitements sémantiques ont souvent besoin de comparer deux synsets ; cela peut être le cas, par exemple, lors de la résolution d'anaphores nominales. De nombreux auteurs ont proposé des définitions de mesures de similarité, et plusieurs implémentations basées sur WordNet sont disponibles. Par exemple, (Pedersen, Patwardhan,

[102] {CHIANTI#1} (*dry red Italian table wine from the Chianti region of Tuscany*).
[103] {DRINK#1, IMBIBE#3} (*take in liquids*) ; {DRINK#2, BOOZE#1, FUDDLE#2} (*consume alcohol*).
[104] http://www.chaumartin.fr/download/wnpolysemy.zip

Michelizzi, 2004) présente plusieurs de ces algorithmes de similarité et une implémentation en Perl appelée WordNet::Similarity.

Nous avons implémenté plusieurs mesures de similarité dans Antelope. De cette façon, un composant de traitement peut choisir celle qui lui semble la mieux adaptée dans un contexte donné. Pour être comparables, les résultats de toutes ces mesures sont ramenés dans l'intervalle [0 ; 1]. La mesure est un nombre réel, valant 1 quand les deux synsets sont identiques, et d'autant plus proche de 0 que les synsets sont différents[105].

(1) Mesure de similarité « structurelle » calculée par parcours du graphe d'hyperonymes

(Lin, 1998) définit la mesure de similarité entre deux synsets *s1* et *s2* avec la formule suivante :

$$sim(s1, s2) = \frac{2.\log P(s)}{\log P(s1) + \log P(s2)}$$

Dans cette formule, *s* est le synset le plus spécifique subsumant *s1* et *s2* dans la hiérarchie de WordNet, et *P(s)* représente le contenu informationnel (Cf. page 51) du synset *s*. Le résultat de cette similarité est dans l'intervalle [1 ; +∞[et vaut 1 quand les deux synsets comparés sont identiques ; il est d'autant plus élevé que les deux synsets sont différents (ou plus précisément, éloignés dans le graphe d'hyperonymie). Nous ramenons cette valeur dans l'intervalle [0 ; 1] en prenant son inverse.

Notre implémentation introduit deux niveaux supplémentaires dans la hiérarchie des verbes. En effet, s'il existe pour les noms une racine unique (ENTITY#1), ce n'est pas le cas pour les verbes ; or, la qualité de la mesure de similarité est fonction de la finesse de la hiérarchie. De façon à rendre tous les verbes comparables, nous avons créé un pseudo-synset qui sert de racine commune à tous les verbes, ainsi que des pseudo-synsets regroupant les catégories lexicales (verbes de mouvement, d'état, etc.).

(2) Mesure de similarité « conceptuelle » calculée par recouvrement des gloses

Cette mesure vectorielle est basée sur le recouvrement des mots entre gloses et utilise une pondération de type TF-IDF. Considérons les deux sens du nom « *samurai* » ; on remarque que les deux définitions ont quatre mots en commun :

- {SAMURAI#1} (*a **Japanese** warrior member of the **feudal military aristocracy***)
- {SAMURAI#2} (***feudal Japanese military aristocracy***)

Le premier synset a pour hyperonyme « personne » et le second « groupe » : ils sont donc très distants (similarité égale à 0,04) du point de vue de la mesure de similarité structurelle. En revanche, ils sont proches (similarité valant 0,56) du point de vue plus « conceptuel » de cette seconde mesure.

b) Regroupement des sens de mots

Comme nous l'avons déjà souligné, WordNet propose un découpage parfois trop fin[106] des sens ; cette caractéristique complexifie la désambiguïsation lexicale. Nous proposons l'application de

[105] Nous ne garantissons pas que les distributions sur [0 ; 1] soient aussi comparables.
[106] Les anglo-saxons parlent de *fine-grained definitions* (par opposition à *coarse-grained definitions*).

plusieurs mesures de similarité pour évaluer la distance entre différents sens d'un même mot et permettre de regrouper les sens très proches. De cette façon, on peut aussi voir WordNet comme un lexique avec des sens macroscopiques.

Nous avons ainsi appliqué les algorithmes de regroupement (voir partie V.C, page 106) aux définitions des sens du verbe EAT, pour les fusionner en sens macroscopiques :

- La figure 19 montre un regroupement effectué avec l'algorithme de Bron-Kerbosch (voir en page 128 la sous-section V.F.4.a). Cet algorithme produit des cliques : on remarquera que le sens EAT#2 apparaît dans les deux premiers groupes.
- La figure 20 montre un autre regroupement du même verbe, utilisant l'algorithme spectral (voir en page 129 la sous-section V.F.4.b). Le résultat de cet algorithme est une partition : chaque sens se retrouve donc dans un groupe et un seul.

Similarity (Composite)

	V#1	V#2	V#3	V#4	V#5	V#6
V#1	1	0,958	0,578	0,145	0,439	0,145
V#2	0,958	1	0,629	0,16	0,474	0,16
V#3	0,578	0,629	1	0,095	0,337	0,095
V#4	0,145	0,16	0,095	1	0,107	0,098
V#5	0,439	0,474	0,337	0,107	1	0,107
V#6	0,145	0,16	0,095	0,098	0,107	1

Clusters (composite)

Cluster #1

1. eat (61) -- (take in solid food; *She was eating a banana; What did you eat for dinner last night?*)
2. eat (13) -- (eat a meal; take a meal; *We did not eat until 10 P.M. because there were so many phone calls; I didn't eat yet, so I gladly accept your invitation*)
3. feed, eat (4) -- (take in food; used of animals only; *This dog doesn't eat certain kinds of meat; What do whales eat?*)

Cluster #2

2. eat (13) -- (eat a meal; take a meal; *We did not eat until 10 P.M. because there were so many phone calls; I didn't eat yet, so I gladly accept your invitation*)
5. consume, eat up, use up, eat, deplete, exhaust, run through, wipe out -- (use up (resources or materials); *this car consumes a lot of gas; We exhausted our savings; They run through 20 bottles of wine a week*)

Cluster #3

4. eat, eat on -- (worry or cause anxiety in a persistent way; *What's eating you?*)

Cluster #4

6. corrode, eat, rust -- (cause to deteriorate due to the action of water, air, or an acid; *The acid corroded the metal; The steady dripping of water rusted the metal stopper in the sink*)

Figure 19 : Regroupement des sens du verbe EAT avec l'algorithme de Bron-Kerbosch

Similarity (Composite)

	V#1	V#2	V#3	V#4	V#5	V#6
V#1	1	0,958	0,578	0,145	0,439	0,145
V#2	0,958	1	0,629	0,16	0,474	0,16
V#3	0,578	0,629	1	0,095	0,337	0,095
V#4	0,145	0,16	0,095	1	0,107	0,104
V#5	0,439	0,474	0,337	0,107	1	0,107
V#6	0,145	0,16	0,095	0,104	0,107	1

Clusters (GlossOverlapping)

Cluster #1

1. eat (61) -- (take in solid food; *She was eating a banana; What did you eat for dinner last night?*)
2. eat (13) -- (eat a meal; take a meal; *We did not eat until 10 P.M. because there were so many phone calls; I didn't eat yet, so I gladly accept your invitation*)
3. feed, eat (4) -- (take in food; used of animals only; *This dog doesn't eat certain kinds of meat; What do whales eat?*)

Cluster #2

4. eat, eat on -- (worry or cause anxiety in a persistent way; *What's eating you?*)

Cluster #3

5. consume, eat up, use up, eat, deplete, exhaust, run through, wipe out -- (use up (resources or materials); *this car consumes a lot of gas; We exhausted our savings; They run through 20 bottles of wine a week*)

Cluster #4

6. corrode, eat, rust -- (cause to deteriorate due to the action of water, air, or an acid; *The acid corroded the metal; The steady dripping of water rusted the metal stopper in the sink*)

Figure 20 : Regroupement des sens du verbe EAT avec l'algorithme spectral

D. Autres ressources à intégrer au lexique sémantique dans le futur

En plus des ressources que nous avons déjà intégrées au lexique sémantique d'Antelope, nous avons identifié d'autres ressources que nous prévoyons d'utiliser prochainement.

1. Lexique de noms déverbaux (NomLex / VerbAction)

NomLex (MacLeod, 1998) est un dictionnaire décrivant le cadre de sous-catégorisation de 1 000 nominalisations en langue anglaise. NomLex précise la correspondance entre les noms déverbatifs et leurs verbes connexes, ainsi que les correspondances entre les arguments verbaux et les positions syntaxiques au sein du groupe nominal. La syntaxe de cette ressource est inspirée de LISP. Ce projet a été repris et étendu dans le cadre de NomBank (Meyers *et al.*, 2004).

Pour le français, une ressource proche dans l'esprit (mais d'ambition plus modeste) est VerbAction, un lexique de noms d'actions morphologiquement apparentés à des verbes, en partie obtenu par acquisition sur le Web (Tanguy, Hathout, 2002).

Les informations de sous-catégorisation des noms déverbatifs font souvent référence à des prépositions dans la description des arguments ; ces derniers sont souvent contraints à utiliser une préposition particulière ou un ensemble de prépositions qui partagent des aspects communs. Il est donc aussi nécessaire de disposer d'un lexique des prépositions.

2. Lexique de prépositions (TPP / PrepLex)

Les prépositions constituent en principe une classe fermée dont on peut énumérer tous les éléments. En pratique, il n'est pas si facile de déterminer leur liste de façon exhaustive.

The Preposition Project (Litkowski, 2002) est un projet conçu pour fournir une caractérisation complète, adaptée au TAL, des sens des prépositions en anglais. 334 prépositions, avec 673 sens, ont été décrites avec un rôle sémantique ou un nom de relation, et une description des propriétés syntaxiques et sémantiques de son complément. Une définition et des exemples d'usage sont donnés pour chaque sens dans TPP.

Une ressource proche pour le français nous semble être PrepLex (Fort, Guillaume, 2007) ; c'est un lexique de prépositions, créé en premier pour fournir des informations à un analyseur syntaxique. On peut aussi citer le projet PrepNet (Saint-Dizier, 2005) qui vise à décrire la syntaxe et la sémantique des prépositions ; ce projet semble néanmoins en être resté à un stade préliminaire (nous n'avons pas trouvé de ressource exploitable).

3. FrameNet

FrameNet (Baker, Fillmore, Lowe, 1998), projet mené à Berkeley à l'initiative de Charles Fillmore, est fondé sur la sémantique des cadres (*frame semantics* en anglais). FrameNet a pour objectif de documenter la combinatoire syntaxique et sémantique pour chacun des sens d'une entrée lexicale à travers une annotation manuelle d'exemples choisis dans des corpus sur des critères de représentativité lexicographique. Les annotations sont ensuite synthétisées dans des tables, qui résument pour chaque mot les cadres avec leurs actants sémantiques et arguments syntaxiques.

FrameNet II compte 825 cadres sémantiques, 10 000 unités lexicales (dont 6 100 complètements annotées) ainsi que 130 000 phrases d'exemples annotés. Les outils et données sont distribués librement. Il existe une correspondance entre les verbes de FrameNet II et ceux de WordNet.

A titre indicatif, voici la description textuelle du cadre "Crime_scenario" :

> A (putative) Crime is committed and comes to the attention of the Authorities. In response, there is a Criminal_investigation and (often) Arrest and criminal court proceedings. The Investigation, Arrest, and other parts of the Criminal_Process are pursued in order to find a Suspect (who then may enter the Criminal_process to become the Defendant) and determine if this Suspect matches the Perpetrator of the Crime, and also to determine if the Charges match the Crime. If the Suspect is deemed to have committed the Crime, then they are generally given some punishment commensurate with the Charges.

Les différents acteurs de ce cadre (*frame elements* en anglais) sont également décrits :

Authorities []	The group which is responsible for the maintenance of law and order, and as such have been given the power to investigate Crimes, find Suspects and determine if a Suspect should be submitted to the Criminal_process.
Charge []	A description of a type of act that is not permissible according to the law of society.

Crime [] An act, generally intentional, that matches the description that belongs to an official Charge.

Perpetrator [] The individual that commits a Crime.

Suspect [] The individual which is under suspicion of having committed the Crime.

Enfin, les cadres sont reliés entre eux par des relations, comme le montre la figure 21.

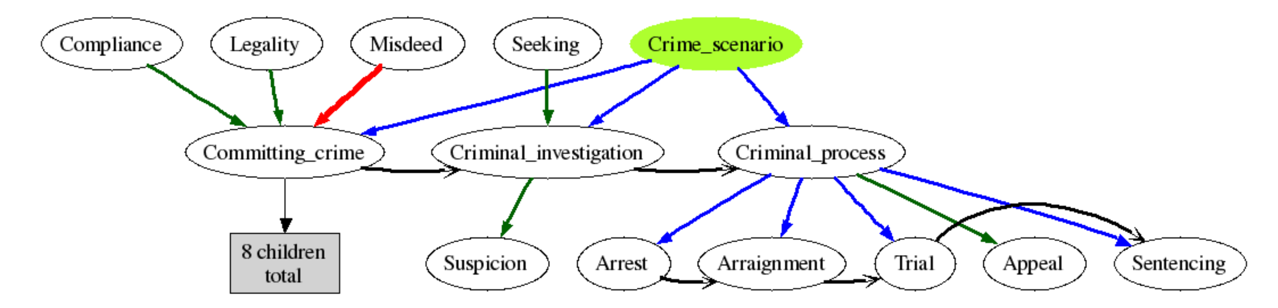


Figure 21 : Exemple de relations entres cadres dans FrameNet

4. Représentation des connaissances à large échelle

Nous avons jusqu'ici présenté des ressources linguistiques. Nous allons à présent glisser de domaine, et passer du TAL à l'intelligence artificielle, ou plus précisément à l'ingénierie des connaissances. Les interactions entre ces deux domaines peuvent avoir lieu dans les deux sens. D'une part, la connaissance du monde permet de lever des ambiguïtés dans de nombreuses tâches de TAL ; disposer d'une large base de données sur le sens commun (par exemple sous forme d'ontologie) permettrait l'injection de connaissances en amont et faciliterait donc l'analyse du texte. D'autre part, de telles bases de connaissances peuvent être amorcées automatiquement en faisant de la fouille de texte (sur des textes encyclopédiques ou règlementaires par exemple) ; mais du fait des imperfections des analyses automatiques, de telles ressources doivent alors obligatoirement être validées manuellement.

La construction de connaissances à large échelle reste donc un verrou scientifique et technologique à lever (usuellement qualifié de *knowledge acquisition bottleneck*). Disposer de ressources à large couverture prêtes à l'emploi peut donc s'avérer intéressant : nous allons en présenter deux ici, l'une constituée manuellement (CYC) et l'autre obtenue par analyse d'un corpus de phrases décrivant des faits (ConceptNet).

a) CYC

CYC (Lenat, 1995) est un projet lancé en 1984 par la société Cycorp. CYC vise à regrouper une ontologie et une base de données complètes sur le sens commun, pour permettre à des applications d'intelligence artificielle d'effectuer des raisonnements similaires à ceux des humains. Cycorp revendiquait déjà en 1995 un investissement de plus de 100 années-homme sur ce projet, sous forme de saisie de faits et de définition d'une axiomatique.

Des fragments de connaissances typiques sont par exemple : « *les chats ont quatre pattes* » ; « *Paris est la capitale de la France* ». Elles contiennent des termes (PARIS, FRANCE, CHAT…) et des assertions

qui relient ces termes entre eux. Grâce au moteur d'inférence fourni avec la base CYC, il est possible d'obtenir une réponse à une question comme « *Quelle est la capitale de la France ?* ».

La base CYC contient des millions d'assertions (faits et règles) rentrées à la main. Elles sont écrites en langage CycL, qui est un langage logique avec une syntaxe proche de celle de LISP. La figure 22 montre par exemple la description d'AbrahamLincoln dans l'interface Web de ResearchCyc.

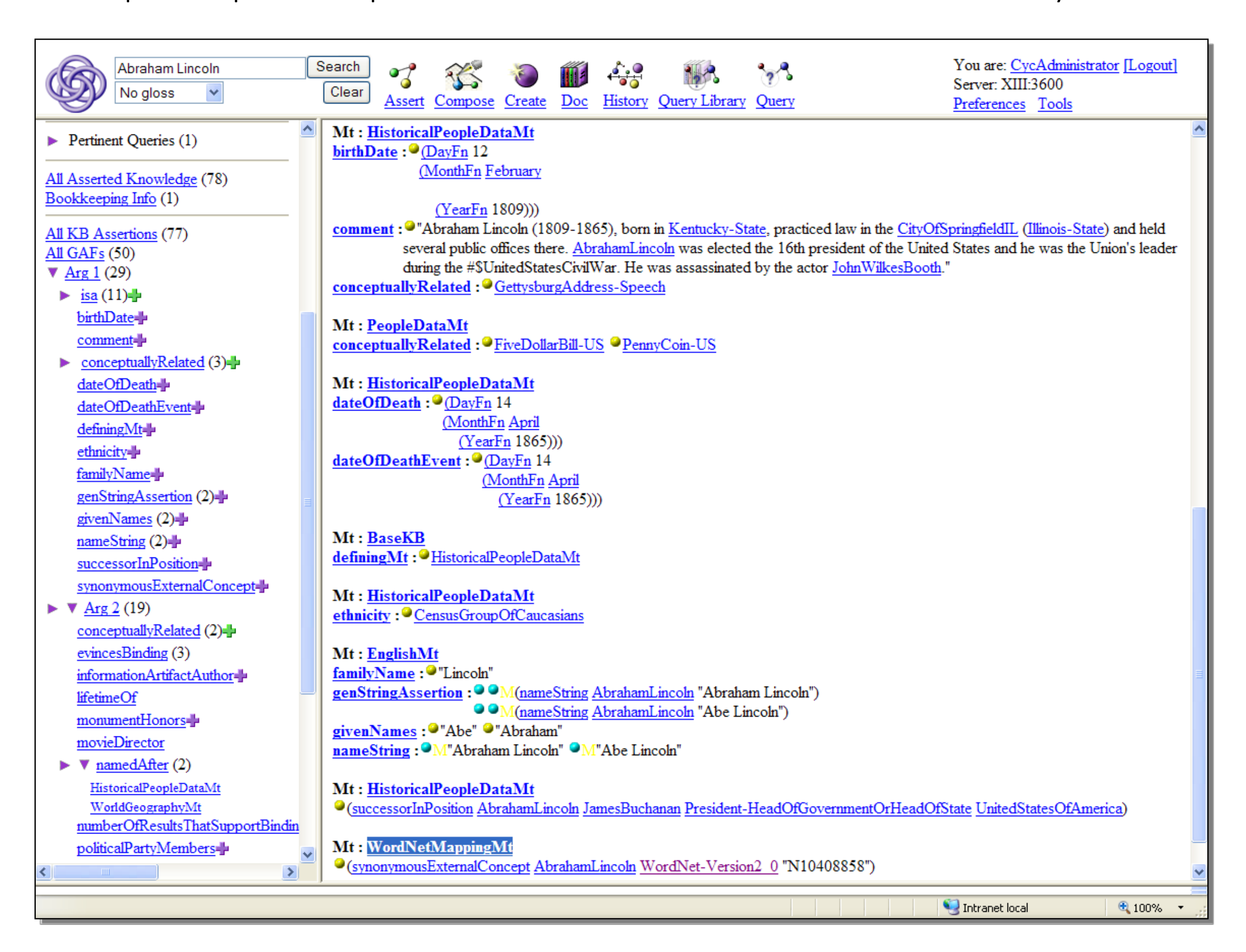


Figure 22 : Interface Web du serveur ResearchCyc

La base de connaissance est divisée en plusieurs milliers de micro-théories (Mt), collections de concepts et faits concernant typiquement un domaine particulier de la connaissance. Une micro-théorie est donc un ensemble d'assertions qui partagent le même point de vue : un domaine particulier, un certain niveau de détail, un certain intervalle de temps, etc. À la différence de la base de connaissance dans son ensemble, chaque micro-théorie doit être exempte de contradictions. Par exemple, Philadelphie était la capitale des Etats-Unis de 1790 à 1800. Dans une micro-théorie couvrant l'intervalle de temps 1790-1800, l'assertion (#$CapitalCity #$UnitedStates #$Philadelphia) sera vraie et, dans une micro-théorie couvrant le XX^{ème} siècle, (#$CapitalCity #$UnitedStates #$Washington) sera également vraie.

ResearchCyc 1.0 est la version réservée au monde de la recherche. Elle compte 300 000 concepts et 3 000 000 d'assertions (faits et règles) utilisant 26 000 relations. Des modules en langage naturel permettent de poser des questions et de rentrer de nouveaux faits sans avoir besoin de connaître

CycL. La version OpenCyc 1.0 est librement accessible, mais ne contient qu'un sous ensemble de ces règles et assertions.

Les deux versions contiennent à ce jour une correspondance partielle entre les concepts de CYC et les synsets de WordNet 2.0. Approximativement 11 300 synsets (8800 noms, 2110 verbes, 330 adjectifs et 35 adverbes) sont liés aux concepts de CYC.

b) ConceptNet

De même que CYC, ConceptNet (Liu, Singh, 2004) est une base de connaissances cherchant à modéliser le sens commun sous forme d'un vaste réseau sémantique. ConceptNet propose aussi un ensemble d'outils permettant d'analyser du texte, pour en extraire des thématiques[107] ou y trouver des analogies[108]. Le réseau sémantique de ConceptNet est un graphe orienté dont les nœuds sont des concepts, et dont les arcs sont des assertions du sens commun portant sur ces concepts. En 2004, il comptait 1,6 millions d'assertions couvrant des aspects spatiaux, physiques, sociaux, temporels et psychologiques de la vie de tous les jours.

A la différence de CYC et WordNet, ressources écrites à la main, ConceptNet a été généré automatiquement à partir de 700 000 phrases du projet OMCS (*Open Mind Common Sense*, mené également au MIT) ; ce projet collaboratif a compté des milliers de contributeurs, sollicités pour écrire de courtes phrases décrivant une situation du quotidien (par exemple « *un livre est fait de papier* », « *le tango est une sorte de danse* », « *on apprend pour connaître plus de choses* »…). Un analyseur syntaxique a été utilisé pour extraire des informations de ces phrases, en utilisant des patrons morphosyntaxiques. Les assertions sont alors exprimées comme des relations entre deux concepts, sélectionnées à partir d'un ensemble fini de relations possibles.

E. Conclusion

Dans cette partie, nous avons présenté un processus de constitution de lexique sémantique à large couverture[109]. Nous avons vu que le cœur de notre lexique contient 117 659 concepts (les synsets de WordNet) auxquels s'ajoutent 300 000 concepts provenant des articles de la Wikipédia (marques, produits, personnes, lieux…). D'autres projets d'ontologies à large échelle (CYC, SUMO…) revendiquent aussi des dizaines ou des centaines de milliers de concepts.

Nous avons donc ici une démarche *top down*, ou démarche descendante, qui vise à constituer *a priori* une représentation du monde aussi exhaustive que possible, sous le prisme des objets linguistiques. Mais cette démarche permet-elle de *tout* couvrir ? Evidemment non. Des concepts nouveaux émergent régulièrement[110] ; de nouveaux termes permettent de désigner ou renommer des concepts déjà existants. Et un lexique, aussi large soit-il, ne permet de couvrir que partiellement toutes les subtilités d'un domaine donné.

---

[107] Par exemple, l'analyse d'un article de presse contenant les concepts « arme à feu », « magasin », « réclamer de l'argent » et « s'échapper » pourrait suggérer les thématiques « vol qualifié » et « crime ».

[108] Par exemple, les concepts « ciseaux », « rasoir », « coupe-ongles » et « épée » sont probablement proches de « couteau » parce qu'ils sont tous <pointus> et peuvent être utilisés pour « couper quelque chose ».

[109] Par ailleurs, le lecteur pourra trouver dans (Cailliau, 2010) plusieurs autres stratégies possibles de gestion de ressources linguistiques.

[110] Par exemple, le métier de *community manager* n'existait pas il y 3 ans ; des marques et sociétés se créent quotidiennement ; de nouveaux produits apparaissent régulièrement, etc.

A l'opposé d'une approche universaliste, une démarche *bottom up*, ou démarche ascendante, consiste à exploiter le corpus que l'on souhaite traiter dans le cadre d'une application. Par exemple, une extraction terminologique permet de découvrir dans un corpus donné les termes, simples ou composés[111], ayant de l'importance ; on peut alors confronter ces termes au lexique de référence, pour éventuellement l'enrichir de nouveaux concepts. De notre point de vue, l'approche pragmatique consiste à mixer ces deux démarches pour disposer du lexique le mieux adapté à une application donnée ; nous préciserons les détails de notre approche d'acquisition de connaissances spécifiques à un domaine (page 136) dans la partie consacrée aux applications de la plate-forme.

[111] Les expressions multi-mots ont l'intérêt d'être plus précises que les termes simples ; par exemple, « numéro de téléphone », « numéro de facture », « numéro de client »… désignent des concepts moins ambigus que « numéro ».

Partie V. Composants de traitement

A. Introduction

1. Composants développés pour la plate-forme

Notre objectif est de fournir une plate-forme facilitant les expériences de TAL. Nous souhaitons donc fournir à l'utilisateur un ensemble de composants de traitement prêts à l'emploi ; l'utilisateur peut alors les assembler rapidement pour les mettre en œuvre dans le cadre d'une application donnée.

Nous présentons dans cette partie la conception des composants d'analyse sémantique que nous avons directement développés pour notre plate-forme. Ces composants, orientés vers l'extraction d'information, s'appuient sur la sortie d'une analyse syntaxique (RSyntS ou RSyntP) et peuvent accéder aux données du lexique sémantique. Au moment de leur implémentation initiale, ces composants fournissaient des résultats à l'état de l'art. Ils sont en cours d'adaptation pour bénéficier de mécanismes d'apprentissage automatique.

2. Introduction à l'extraction d'information

Les composants d'analyse sémantique de notre plate-forme traitent des tâches au niveau :

- De la phrase : reconnaissance d'entités nommées (page 93), extraction de relations (page 106) et analyse d'opinion et de sentiments (page 115).
- Du document : résolution d'anaphores et de coréférences (page 122).
- Du corpus : regroupement de documents (page 125).

Comme on le voit, il s'agit de tâches d'extraction d'information, que (Moens, 2006) définit comme étant « *l'identification, effectuée suite ou simultanément à une classification et structuration en classes sémantiques, d'informations spécifiques trouvées dans des sources de données non structurées, telles que du texte en langage naturel, fournissant des aides supplémentaires aux systèmes d'information pour accéder et interpréter ces données non structurées* »[112]. L'extraction d'information ne cherche pas à attribuer une valeur de vérité aux informations extraites. Elle vise à reconnaître des entités et des relations entre entités au sein d'une phrase ou d'un paragraphe, sans prétendre fournir une compréhension globale de l'information contenue dans un document. (Tannier, 2006) fournit un panorama des techniques de traitement automatique du langage naturel utilisées en extraction et recherche d'informations.

Les premiers systèmes d'extraction d'information ayant une importance historique sont FRUMP (DeJong, 1982) et FASTUS (Hobbs *et al.*, 1996). Plusieurs campagnes d'évaluation d'envergure ont permis de mesurer les progrès dans ces domaines. On peut citer, pour l'anglais, les conférences MUC

---

[112] "*The identification, and consequent or concurrent classification and structuring into semantic classes, of specific information found in unstructured data sources, such as natural language text, providing additional aids to access and interpret the unstructured data by information systems*". (C'est notre traduction.)

(*Message Understanding Conference*) qui se sont déroulées de 1987 à 1998 (Grishman, Sundheim, 1996) sous l'égide du DARPA[113] puis ACE (*Automatic Content Extraction*), menées de 2000 à 2008 (Doddington *et al.*, 2004). En France, les campagnes d'évaluation Amaryllis (1997, 1999) et ESTER (transcription d'une centaine d'heures de nouvelles orales) ont permis des évaluations sur le français.

MUC7 (1998) portait notamment sur l'identification des noms propres dans des textes journalistiques. Cette tâche est actuellement celle qui obtient les meilleures performances en extraction d'information, avec des scores approchant un jugement humain. Sur des corpus journalistiques, les scores obtenus (moyenne harmonique combinant précision et rappel) sont proches de 90 % ; les particularités d'une langue (notamment en ce qui concerne l'usage des majuscules) peuvent faire varier ces résultats.

L'adoption progressive par les projets industriels des standards émergents du Web sémantique (Feigenbaum *et al.*, 2007) a récemment accentué l'intérêt pour cette tâche. Dans ce contexte, elle permet d'associer des métadonnées à un texte[114] pour en améliorer l'indexation par un moteur de recherche.

3. Des systèmes de règles à l'apprentissage automatique

L'évolution la plus significative que nous avons perçue en TAL, depuis le début de nos travaux, concerne l'importance grandissante des mécanismes d'apprentissage automatique (*machine learning* en anglais). Ce champ d'étude de l'intelligence artificielle vise à découvrir automatiquement les corrélations présentes dans un jeu de données afin d'en extraire les connaissances. Cela revient donc à calculer les paramètres d'un modèle en s'assurant de sa validité.

a) Intérêts de l'apprentissage

L'approche classique en TAL consiste à créer un modèle linguistique symbolique avec un système de règles écrites manuellement. Mais un tel modèle est lourd à mettre en place : il requiert des connaissances pointues en linguistique et demande un investissement coûteux en temps humain ; on parle de goulet d'étranglement dans l'acquisition des connaissances (*knowledge acquisition bottleneck*). La correction d'erreur, la maintenance d'un tel modèle et le passage à d'autres langues s'avèrent donc problématiques.

L'apprentissage automatique permet de résoudre les problèmes qu'il est difficile, voire impossible, d'aborder par des moyens algorithmiques plus classiques, quand l'explicitation des règles est trop complexe ou débouche sur une explosion combinatoire. L'apprentissage automatique a progressivement concerné un grand nombre de tâches de TAL, allant de l'étiquetage morphosyntaxique à la classification, en passant par la fouille de texte ou encore l'analyse syntaxique probabiliste.

L'intérêt d'un modèle fondé sur l'apprentissage automatique est, en effet, de pouvoir être mis en place puis facilement adapté à de nouveaux domaines, pour peu que des corpus annotés soient disponibles. Cette approche est aussi intéressante pour concevoir des systèmes de TAL largement indépendants d'une langue donnée.

---

[113] *Defense Advanced Research Projects Agency*, ou Agence pour les Projets de Recherche Avancée de Défense, l'agence américaine chargée de la R&D des nouvelles technologies destinées à un usage militaire.

[114] La description des entités nommées et des relations peut se faire, par exemple, sous forme de triplets RDF.

b) Importance grandissante de l'apprentissage en TAL

L'utilisation de l'apprentissage automatique en TAL n'est certes pas nouvelle, mais sa progression est très nette en une décennie. La figure 23 montre l'évolution, entre 1998 et 2011, du pourcentage des articles d'ACL mentionnant le terme *machine learning*[115]. On voit qu'on est passé en 14 années de moins de 10 % à près de 30 %.

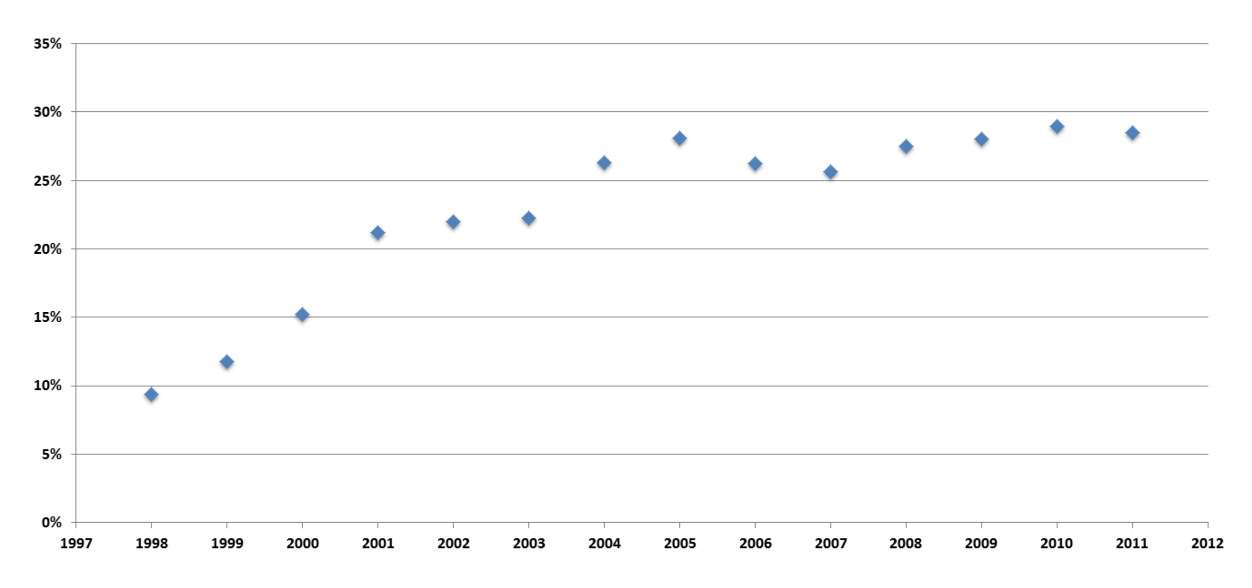


Figure 23 : Progression entre 1998 et 2011 des articles d'ACL mentionnant « *machine learning* »

c) Différents modes d'apprentissage

Il existe un grand nombre d'algorithmes d'apprentissage. On peut citer les réseaux de neurones, la méthode des *k* plus proches voisins, les arbres de décision, la classification bayésienne, les HMM (réseaux cachés de Markov), les SVM (*Support Vector Machine*) et les CRF (*Conditional Random Fields*). Une introduction de référence à ces différents algorithmes peut être trouvée dans (Cornuéjols, Miclet, Kodratoff, 2002).

Il faut donc être en mesure de choisir celui qui est le mieux adapté à une tâche particulière. Il existe différents modes d'apprentissage, faisant ou non intervenir une validation humaine en cours de processus.

(1) Apprentissage supervisé

L'apprentissage supervisé a pour objectif de produire des règles à partir d'un corpus préalablement annoté contenant des exemples catégorisés du phénomène que l'on souhaite apprendre. La mise en œuvre pratique de techniques d'apprentissage supervisé se heurte à différents types de difficultés :

- Il faut disposer d'un corpus d'apprentissage suffisamment représentatif pour garantir l'exhaustivité de la base d'apprentissage. Ce problème est particulièrement sensible pour l'apprentissage en TAL, car le langage humain n'est pas régulier et les règles générales souffrent de nombreuses exceptions (pluriel des mots se terminant en -ou, par exemple).
- Préalablement à l'apprentissage, le corpus doit être annoté selon la catégorie d'information que l'on souhaite apprendre ; chaque exemple est explicitement associé à une étiquette. Procéder manuellement à l'annotation d'un corpus volumineux représentatif est coûteux.

---

[115] Pour établir ce graphique, nous avons utilisé l'ACL Anthology Searchbench (http://aclasb.dfki.de), en calculant, pour chaque année, le ratio entre le nombre total d'articles et ceux citant « *machine learning* ».

- Il faut guider un système d'apprentissage en lui indiquant les caractéristiques discriminantes du phénomène que l'on souhaite apprendre (par exemple, que le pluriel d'un nom français peut être indiqué par le suffixe –s ou -x) ; il faut donc une expertise linguistique.

(2) Apprentissage non-supervisé

On parle d'apprentissage non-supervisé dans le cas où on ne dispose que d'exemples, mais non d'étiquettes, et que le nombre ou la nature des classes n'est pas déterminée à l'avance. Le corpus est donc fourni brut, sans annotation. L'algorithme est censé expliciter tout seul la structure plus ou moins cachée de données hétérogènes, en les organisant en sous-groupes où les données similaires sont regroupées d'une façon homogène. Un humain expert du domaine peut alors éventuellement attribuer manuellement une classe à chacun de ces sous-groupes. Le partitionnement d'un corpus en groupes de documents similaires (*clustering*) est un exemple d'apprentissage non-supervisé.

(3) Apprentissage semi-supervisé

L'apprentissage semi-supervisé utilise conjointement un ensemble de données étiquetées et non-étiquetées. L'intérêt est d'une part d'améliorer significativement la qualité ou la rapidité de l'apprentissage (Blum, Mitchell, 1998) et d'autre part de nécessiter moins de temps d'annotation des corpus.

d) Utilisation dans le cadre de notre plate-forme

Les composants d'Antelope étaient initialement tous définis par des systèmes de règles codées manuellement. La reconnaissance d'entités nommées et le regroupement de documents ont évolué pour intégrer des techniques d'apprentissage automatique, à la place ou en complément des implémentations précédentes. L'introduction de ces techniques a permis une amélioration des performances et d'obtenir des comportements plus robustes en améliorant le rappel. Notre objectif est de progressivement généraliser cette approche à l'ensemble des traitements sémantiques pour en améliorer les performances.

e) Difficultés rencontrées

L'apprentissage automatique porte des promesses importantes en TAL. Soulignons toutefois que sa mise en œuvre effective n'a rien de simple. En effet, le cadre mathématique sous-jacent est généralement complexe. Il fait appel à des connaissances ou talents généralement nouveaux pour le praticien du TAL, en plus des aspects linguistiques ou informatiques.

Notre expérience personnelle est qu'on peut certes commencer à utiliser des tels composants de calcul en tant que boîte noire, sans chercher à en comprendre le fonctionnement interne. Néanmoins, une compréhension minimale des algorithmes sous-jacents est importante pour bien choisir celui qui convient pour un problème particulier.

L'autre point concerne la performance des implémentations ; la complexité des algorithmes nécessite souvent des heures (voire des jours) de calcul[116]. Il est donc parfois nécessaire de les réimplémenter en les optimisant, ce qui peut devenir compliqué. Nous explorons l'approche consistant à exploiter la structure hautement parallèle du processeur des cartes graphiques[117].

---

[116] Par exemple, des techniques d'apprentissage itératives telles que les descentes de gradient.

[117] Un GPU est efficace sur la tâche de rendu 3D, mais peut aussi servir aux algorithmes d'apprentissage.

B. Reconnaissance d'entités nommées

Ce chapitre présente la détection d'entités nommées, que nous effectuons avec une technique duale, mixant système de règles et apprentissage automatique. Nous utilisons des champs conditionnels aléatoires (ou CRF –*Conditional Random Fields*– en anglais). Après une introduction au concept de CRF, nous présentons succinctement le fonctionnement du détecteur d'entités nommées de Stanford, puis celui de notre propre composant et des caractéristiques qu'il utilise pour l'apprentissage. Nous l'avons utilisé avec succès sur plusieurs projets, avec des documents contenant des entités de types très différents, comme des dépêches de l'AFP (Cf. le projet SCRIBO, page 132), des avis de consommateurs (voir page 141) et des documents RH (voir page 146).

1. Introduction

Une entité nommée est une unité linguistique qui désigne un élément précis de l'univers du discours. Cela peut-être un nom propre (« Picasso », « France »), ou un ensemble de mots (« le Président de la République »). Les entités nommées désignant le plus souvent les éléments sur lesquels portent le discours, leur détection est donc essentielle dans les applications d'extraction ou de recherche d'informations textuelles.

a) Versatilité des types d'entités

Les entités nommées dénotent des éléments de natures très différentes. En fonction de la nature de la tâche à réaliser, les types d'entités qu'on cherche à détecter varient fortement :

- Une application grand public comme Skype ne cherche à identifier qu'un seul type d'entité, les **numéros de téléphone**, pour faciliter leur numérotation. Skype est robuste dans cette tâche, en tenant compte d'une large combinatoire de variantes de surface possibles (présence ou non de parenthèses, d'espaces, d'un indicatif international ou régional, etc.) et en sachant exclure les numéros de fax (quand une telle information existe explicitement).
- Une application spécialisée dans la chimie cherche typiquement à reconnaître des **molécules**, écrite sous forme de formule (« CH_4 ») ou en toutes lettres (« méthane »).
- L'analyse de dépêches de presse vise à faire ressortir des **personnes**, **lieux** ou **organisations**.
- Dans un avis de consommateur, on cherche plutôt à identifier les **produits**, **marques** et **concurrents** cités, ainsi que les **opinions** exprimées.
- L'analyse d'un CV est plutôt centrée sur l'extraction des **compétences**, **talents**, **expériences passées**, **langues** parlées, etc. du candidat.

OpenCalais est un service d'annotation en ligne proposé depuis 2008 par ClearForest, une filiale de Thomson Reuters. Il identifie dans du texte des entités nommées et des relations avec l'approche décrite dans (Feldman *et al.*, 2001). OpenCalais associe une annotation en RDF à chaque entité, fait et événement détecté ; l'exemple suivant illustre la reconnaissance de la société Gazprom dans un article de presse :

```
<rdf:Description rdf:about="http://d.opencalais.com/comphash-1/5d25b012-282d-3ace-9299-4b7b144cde9f">
  <rdf:type rdf:resource="http://s.opencalais.com/1/type/em/e/Company" />
  <c:name>Gazprom</c:name>
  <c:nationality>N/A</c:nationality>
</rdf:Description>
```

Dans un texte anglais, OpenCalais identifie une quarantaine de types d'entités nommées[118] ainsi que 77 types de relations ou de faits entre ces entités[119]. Notons que les sociétés et lieux géographiques sont, de plus, correctement désambiguïsés en général (par rapport à DBPedia, Cf. IV.A.2.d). En français, le service rendu est plus modeste et ne détecte actuellement que 15 classes d'entités[120].

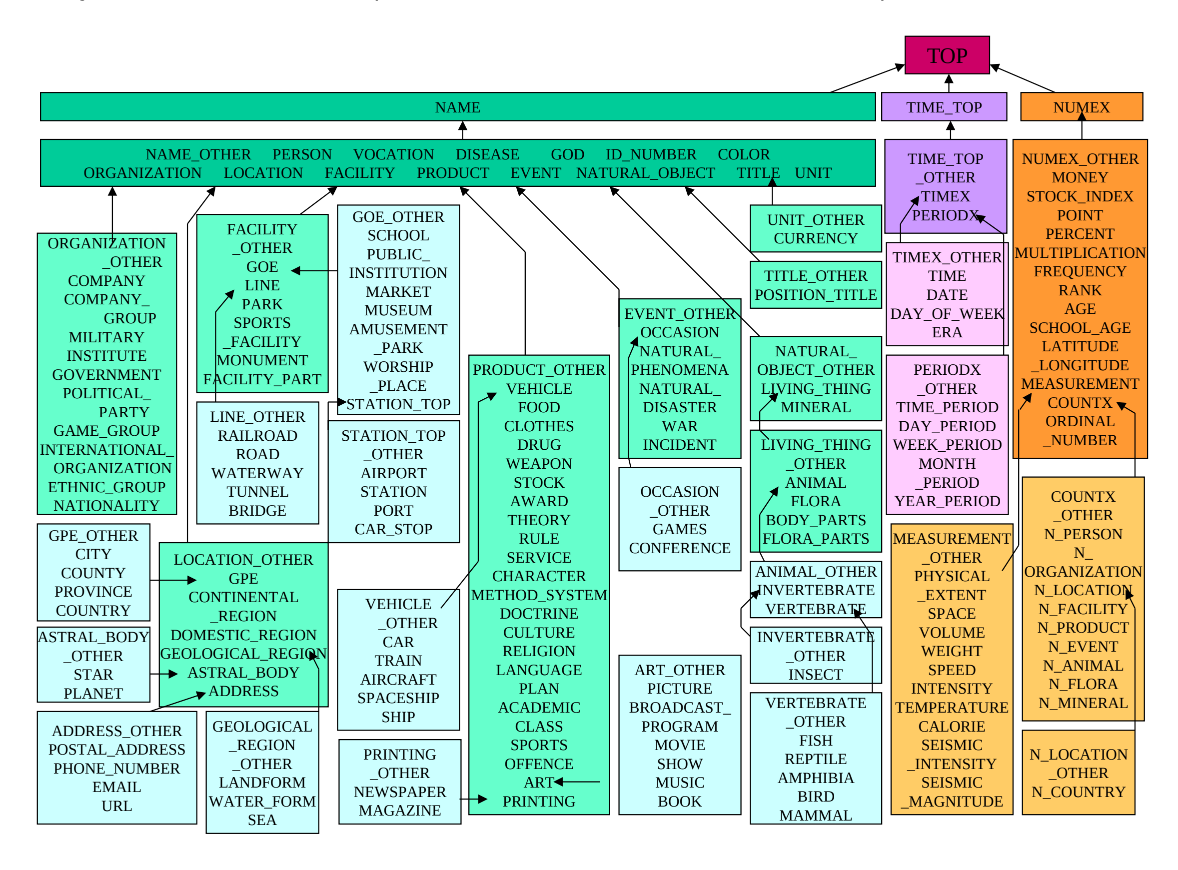


Figure 24 : Hiérarchie d'entités nommées (version 6.1.2) proposée par (Sekine *et al.*, 2002)

[118] Anniversary, City, Company, Continent, Country, Currency, EmailAddress, EntertainmentAwardEvent, Facility, FaxNumber, Holiday, IndustryTerm, MarketIndex, MedicalCondition, MedicalTreatment, Movie, MusicAlbum, MusicGroup, NaturalFeature, OperatingSystem, Organization, Person, PhoneNumber, PoliticalEvent, Position, Product, ProgrammingLanguage, ProvinceOrState, PublishedMedium, RadioProgram, RadioStation, Region, SportsEvent, SportsGame, SportsLeague, Technology, TVShow, TVStation, URL.

[119] Acquisition, Alliance, AnalystEarningsEstimate, AnalystRecommendation, Arrest, Bankruptcy, BonusSharesIssuance, BusinessRelation, Buybacks, CompanyAccountingChange, CompanyAffiliates, CompanyCompetitor, CompanyCustomer, CompanyEarningsAnnouncement, CompanyEarningsGuidance, CompanyEmployeesNumber, CompanyExpansion, CompanyForceMajeure, CompanyFounded, CompanyInvestment, CompanyLaborIssues, CompanyLayoffs, CompanyLegalIssues, CompanyListingChange, CompanyLocation, CompanyMeeting, CompanyNameChange, CompanyProduct, CompanyReorganization, CompanyRestatement, CompanyTechnology, CompanyTicker, CompanyUsingProduct, ConferenceCall, ContactDetails, Conviction, CreditRating, DebtFinancing, DelayedFiling, DiplomaticRelations, Dividend, EmploymentChange, EmploymentRelation, EnvironmentalIssue, EquityFinancing, Extinction, FamilyRelation, FDAPhase, IndicesChanges, Indictment, IPO, JointVenture, ManMadeDisaster, Merger, MovieRelease, MusicAlbumRelease, NaturalDisaster, PatentFiling, PatentIssuance, PersonAttributes, PersonCareer, PersonCommunication, PersonEducation, PersonEmailAddress, PersonRelation, PersonTravel, PoliticalEndorsement, PoliticalRelationship, PollsResult, ProductIssues, ProductRecall, ProductRelease, Quotation, SecondaryIssuance, StockSplit, Trial, VotingResult.

[120] City, Company, Continent, Country, Currency, EmailAddress, FaxNumber, MarketIndex, NaturalFeature, Organization, Person, PhoneNumber, ProvinceOrState, Region, URL

OpenCalais a été, à notre connaissance, le premier service en ligne gratuit et performant d'extraction d'information. C'est aussi la preuve qu'un passage à l'échelle est possible sur ces tâches.

(Sekine *et al.*, 2002) propose une hiérarchie d'entités nommées qui contient approximativement 150 types, comme il est montré en figure 24. Une telle hiérarchie est-elle exhaustive ? La réponse est clairement négative, car on peut toujours la raffiner en fonction de la tâche exacte à réaliser.

Une analyse d'avis de consommateurs ne se contentera généralement pas de détecter des produits alimentaires (FOOD), mais identifiera plus finement les références aux fruits, légumes, fromages, etc. Un système spécialisé dans les fromages spécialisera la reconnaissance en fonction de leur origine ou de leur type (pâte crue, pâte cuite…). On voit donc que la reconnaissance d'entités nommées a une dimension fractale et qu'en fonction de l'application visée, on cherchera à reconnaître des types très généraux (personnes, lieux…) ou au contraire arbitrairement fins.

b) Ambiguïtés à résoudre

La caractérisation de la nature de l'entité nommée est tout aussi essentielle que sa détection. Toutefois, ceci peut se révéler une tâche délicate, car un même mot, tel que « Paris », peut endosser différentes classes selon le contexte :

- « Il vit à **Paris** » (Ville).
- « **Paris** se révolte contre la hausse de prix des horodateurs » (Les habitants de Paris).
- « **Paris** n'a pas accepté les exigences de Bruxelles » (Le gouvernement français).

Le mot « Paris » peut aussi apparaître comme partie d'une expression multi-mots ou d'une entité nommée, comme dans :

- « C'est **Paris** Hilton » (prénom d'une Personne).
- « On lit **Paris Match** » (Magazine).

Le même problème se pose sur les noms de marques qui sont aussi des noms communs[121], à l'initiale en majuscule près. L'analyse d'avis de consommateurs nécessite une désambiguïsation du vocable correspondant pour être sûr qu'il exprime effectivement un avis sur la marque.

2. Détection en utilisant des gazettes contextuelles

Le système le plus simple de reconnaissance d'entités nommées consiste à les chercher à l'intérieur d'une simple liste à plat de termes établie pour chaque classe à reconnaître. Dans la suite, nous appelerons *gazette* une telle liste (par analogie avec le terme *gazetteer* employé dans la littérature anglo-saxonne[122]). Les exemples précédents montrent que se fier à une solution naïve de ce type est trop restrictif. Il faut donc mettre en œuvre des solutions plus sophistiquées.

La difficulté réside dans le fait de donner assez d'information contextuelle pour lever les éventuelles ambiguïtés. De façon à permettre de les lever, nos gazettes contextuelles associent à chaque entrée des éléments contextuels de désambiguïsation sous la forme de termes activateurs de sens.

---

[121] Par exemple Orange, Total, Boulanger, Carrefour, Air Liquide, Ciel, Casino, etc. Une ambiguïté de ce type existe dans un tiers des noms de sociétés du CAC40.

[122] Une traduction plus usuelle pourrait être *nomenclature* ou *répertoire* ou *index géographique*.

Notre démarche est alors :

- En premier lieu, identifier les termes potentiellement ambigus : par exemple, ORANGE.
- Identifier les expressions multi-mots contenant « orange » mais qui correspondent à des entités distinctes : JUS D'ORANGE, SIROP D'ORANGE, CONFITURE D'ORANGE, NECTAR D'ORANGE, CANARD A L'ORANGE, SORBET A L'ORANGE…
- Ensuite, pour chaque terme ambigu, chercher les différents sens intéressants à reconnaître dans le contexte : sur des avis de consommateurs dans l'univers de la grande distribution, on peut chercher à distinguer les sens d'ORANGE#1$_{[fruit]}$ et ORANGE#2$_{[marque\ télécom]}$ en renonçant à reconnaître ORANGE#3$_{[couleur]}$ et ORANGE#4$_{[ville]}$ (si on estime que la probabilité d'apparition de ces sens dans le corpus est marginale).
- Enfin, énumérer pour chaque sens ciblé les termes qui seront fréquemment en co-occurrence dans un contexte local (tel qu'une fenêtre de mots) et qui joueront le rôle d'activateur de sens, éventuellement en association avec des contraintes morphosyntaxiques.

En ce qui concerne ce dernier point, dans notre exemple, on obtiendra :

- ORANGE#1$_{[fruit]}$: fruit, pressée, kg, sanguine, écorce, salustiana, filet, maltaise, citron, manger, déguster… Un déterminant comme « des », « les » ou « une » juste à gauche permet aussi d'activer ce sens.
- ORANGE#2$_{[marque\ télécom]}$: internet, mobile, mobicarte, téléphone, abonnement, contrat, télécom, opérateur, sfr, bouygues, messagerie, wanadoo, boutique, livebox, sim, désimlockage, résiliation, sms, tv… Une préposition comme « chez » juste à gauche permet aussi d'activer ce sens.
- Il faut aussi énumérer les termes qui activeront les sens qu'on ne souhaite pas reconnaître, en jouant le rôle d'inhibiteurs de sens : couleur, ville, code postal d'Orange (84100), théâtre… Le sens ORANGE#3$_{[couleur]}$ peut être reconnu si c'est un adjectif ; une préposition spatiale comme « à » juste à gauche permet d'activer le sens ORANGE#4$_{[ville]}$ (« j'habite à Orange ») ou éventuellement ORANGE#2$_{[marque\ télécom]}$ (« il est abonné à Orange »).

En pratique, un tel système donne déjà des résultats satisfaisants dans le contexte applicatif (relativement fermé) du système d'analyse d'avis de consommateurs présenté au chapitre VI.D (page 141). Voici quelques extraits de verbatim où le sens ORANGE#1$_{[fruit]}$ est reconnu uniquement grâce au système de gazettes contextuelles :

- Le 3 janvier vers 16h30, rayon fruits et légumes vide. Plus d'oranges.
- J'ai acheté le 21.12.10 vers 11h un filet de 3kg d'oranges à déguster (étiquette et ticket de caisse joint) 3.49€ or le lendemain je constate que 2 fruits sont pourris et immangeables.
- Les oranges en promotion super ! à quand à nouveau ?
- En l'espace d'un mois seulement 2 promotions sur des oranges.
- Dommage que vous n'ayez pas de fruits, oranges et citrons non traités en cette période où chaque ménage alsacien prépare des petits gâteaux de Noël et autres recettes.
- Chaque semaine le prix affiché des oranges est différent en caisse lors du passage.
- HIER LES ORANGES 10 KGS ETAIENT A 5.99€ AUJOURD HUI A 7.49€. HONTEUX !
- Dommage que vous n'ayez que des oranges et mandarines venant d'Espagne même bio.

De même, voici quelques exemples de verbatims où ORANGE#2$_{\text{[marque télécom]}}$ a été correctement reconnu par le même système :

- Ouverture de ligne pour une cliente a déjà un forfait internet chez Orange.
- Je suis passé hier dans votre magasin pour changer mon téléphone chez orange.
- Le client attend toujours le remboursement suite à OFFRE PROMOTIONNELLE orange livebox.
- CHANGEMENT DE TELEPHONE PORTABLE SUITE ENVOI SMS ORANGE.
- Je n'ai toujours pas de décodeur TV fourni par orange.
- QUAND NOUS NE VENONS PAS CHEZ VOUS NOUS ALLONS A LA BOUTIQUE ORANGE.
- Le client n'a pas pu acheter une recharge mobicarte orange 20€ (édition spéciale).
- Je fais suite à mon dernier mail concernant mon abonnement à orange net plus.
- VOTRE VENDEUR M'A DIT QU'IL S'OCCUPAIT DE LA RESILIATION DE MON OPERATEUR PRECEDENT OR DEPUIS JE M'APERCOIS QUE JE RECOIS ENCORE DES FACTURES ORANGE.
- J'ai besoin des coordonnées de l'acheteur pour le désimlockage du téléphone chez Orange.

3. Les champs conditionnels aléatoires (CRF)

Le système symbolique que nous venons de présenter repose sur des gazettes contextuelles avec des termes activateurs ou inhibiteurs de sens. S'il donne de premiers résultats satisfaisants, il ne fonctionne qu'avec un nombre fini de cas, qui doivent être explicitement prévus ; il n'est pas capable de généralisation, et ne saura reconnaître de nouvelles instances d'entités si elles sont absentes des gazettes. Or, dans la plupart des domaines, de nouvelles entités apparaissent régulièrement (produits et marques dans un contexte dans la grande distribution, sociétés et personnes dans les articles de presse, par exemple). L'apprentissage automatique permet de dépasser ces limites.

Le type d'apprentissage automatique que nous utilisons pour la détection d'entités nommées repose sur les champs conditionnels aléatoires (CRF). Ces modèles probabilistes ont été proposés par (Lafferty, McCallum, Pereira, 2001) afin de pallier certains défauts des modèles utilisés jusqu'alors dans le domaine du traitement et de l'annotation de séquences de données. Ils ont rapidement montré leur intérêt dans l'étiquetage morphosyntaxique et la détection d'entités nommées[123] ; nous nous en servons d'ailleurs pour effectuer ces deux tâches. Une introduction à leur utilisation peut être trouvé dans (Wallach, 2004) ou (Sutton, McCallum, 2006). Des extensions (XCRF) ont été proposées plus récemment (Jousse *et al.*, 2006) pour effectuer des annotations sur des arbres, et pas seulement sur des structures linéaires.

Un modèle basé sur les CRF est un modèle probabiliste, comme les réseaux bayésiens, les chaînes cachées de Markov, ou les modèles à entropie maximale. Tous les modèles probabilistes permettant l'annotation de séquences reposent sur des principes similaires. On peut notamment les voir comme des variations des réseaux de Markov à états cachés.

De façon grossière, on peut dire qu'un tel modèle, appliqué à l'étiquetage morphosyntaxique ou la détection d'entités nommées, va calculer la probabilité pour qu'un mot appartienne à une classe donnée, puis lui associer la classe maximisant cette probabilité. Pour ne pas introduire un biais et ne proposer que les étiquettes les plus probables à chaque fois, on utilise des modèles capables de

[123] Voir par exemple (McCallum, Li, 2003) pour la reconnaissance d'entités nommées en anglais et (Zidouni, Glotin, Quafafou, 2009) en français.

modéliser les dépendances entre les observations (les mots) et les classes associées : les CRF. L'annexe I.C (page 203) rappelle les bases mathématiques sous-jacentes.

4. Découverte des CRF

a) Présentation du Stanford NER

Nous n'avions pas d'expérience pratique de l'apprentissage automatique avant d'aborder la tâche de reconnaissance des entités nommées. Nous avons pris le parti de défricher ce domaine en utilisant un composant existant, le Stanford Named Entity Recognizer (Stanford NER dans la suite), développé en Java par le NLP Group de l'Université de Stanford.

Ce programme permet l'étiquetage de séquences de mots en classes d'entités nommées. Il implémente une version générale des CRF ainsi que de nombreuses caractéristiques adaptées à la reconnaissance d'entités nommées ; il est conçu initialement pour la détection d'entités nommées dans des textes en anglais.

b) Démarche adoptée

Nous avons procédé à une utilisation en boîte noire (utilisation de la documentation et des exemples fournis), puis en boîte blanche (par examen du code source), du Stanford NER ; cela nous a permis de nous familiariser rapidement avec les techniques d'apprentissage automatique, ainsi qu'avec la définition des caractéristiques adaptées à la tâche de détection d'entités nommées.

Nous avons ensuite étudié l'adaptation de ce programme à des corpus en français. Pour cela, il fallait déterminer les éléments spécifiques à l'anglais présents dans le programme, puis les adapter afin qu'ils fonctionnent également pour le français. L'examen du code source du nous a montré que certains éléments étaient difficilement utilisables avec une langue autre que l'anglais. On retrouve par exemple des patrons morphosyntaxiques visant à reconnaître des titres honorifiques, des dates ou encore des éléments ordinaux (« *first* », « *eleventh* »…) spécifiques à l'anglais. Il existe également une classe chargée de remplacer certains suffixes typiquement britanniques (tels que -*ise*) par leur équivalent américain (-*ize*).

c) Performances annoncées

Le NLP Group de Stanford annonce comme F-score de reconnaissance : 89,19 % sur la classe PERSON ; 80,15 % sur la classe ORGANIZATION ; 85,48 % sur la classe LOCATION. Néanmoins, nous n'avons pas réussi à reproduire ces résultats, faute d'avoir réussi à mettre en œuvre l'ensemble des caractéristiques et options permettant de l'atteindre.

d) Notre bilan

Il nous semble que le Stanford NER n'a pas été conçu dans l'optique d'être personnalisé sans en modifier directement le code source. Son architecture rend difficile la compréhension de certains de ses mécanismes ; hériter des classes existantes apparaît relativement complexe. Il est également difficile d'introduire des options nouvelles, et un manque de documentation rend complexe l'utilisation des options existantes : certaines sont présentes dans le code mais non documentées ; d'autres sont documentées mais absentes du code source, ou ont déclenché des erreurs lors de nos tentatives d'exécution. Nous n'avons au final pas réussi à utiliser certaines caractéristiques que nous aurions souhaité tester (gazettes, étiquettes morphosyntaxiques).

Notre bilan est que le Stanford NER n'est pas facilement utilisable en tant que bibliothèque de code extensible. De plus, nous souhaitions utiliser les analyseurs linguistiques dont nous disposons dans Antelope ; nous avons donc décidé de mettre en œuvre une autre bibliothèque de CRF. Néanmoins, les résultats obtenus ici nous ont servi de référence et d'éléments de comparaison.

5. Notre implémentation

a) Présentation

Après un examen des bibliothèques de code *open source* gérant des CRF, notre choix s'est porté sur une implémentation écrite en Java pour l'annotation de données séquentielles (Sarawagi, Cohen, 2004). Cette bibliothèque de code CRF a été conçue pour être réutilisée par d'autres programmes, en étendant son comportement d'origine. Il est relativement aisé de travailler avec tout type de données, à condition d'écrire un adaptateur spécifique[124].

b) Annotation d'un corpus d'apprentissage

Afin d'entrainer le module d'apprentissage par CRF, il faut lui fournir une quantité de données d'entraînement « suffisante », sous forme d'un corpus pré-annoté. Nous disposions pour ce faire de deux corpus de dépêches, l'un en anglais et l'autre en français, annotés avec l'outil OpenCalais. Ces corpus se présentaient sous la forme de fichiers au format RDF/XML. Nous avons procédé à une correction manuelle (sans garantie d'exhaustivité) de ce premier corpus.

A l'issue de ce prétraitement, nous disposons d'un corpus d'entraînement annoté. Les entités nommées ne se résument pas forcément à un mot isolé. Par exemple, un nom de personne se compose généralement d'un prénom suivi d'un nom de famille, et peut être introduit par un titre (Monsieur, Mme, président…) ; un nom de société est éventuellement suivi par sa forme juridique (SARL, SAS, GmbH, Ltd…). Nous avons donc annoté les entités nommées en utilisant la convention IOB (pour *inside*, *outside*, *begin*) proposée initialement par (Ramshaw, Marcus, 1995) pour annoter les chunks. Le préfixe « B_ » ou « I_ » est ajouté selon que le terme courant est le début ou non de l'entité nommée ; lorsque le mot courant ne représente aucune entité, on lui associe l'étiquette « O ». Avec cette convention, on annote par exemple « le président Sarkozy a décidé… » de la façon suivante : « le/O président/B_PERSONNE Sarkozy/I_PERSONNE a/O décidé/O… ».

Notons que d'autres schémas d'annotations plus ou moins complexes sont évidemment possibles. Par exemple, le guide d'annotation du projet Quaero (Rosset *et al.*, 2011) présente en détail les principes ayant servi à étiqueter des corpus de presse écrite et orale (trois millions de mots en tout).

c) Définition des caractéristiques

Les caractéristiques (ou *features* en anglais) sont déterminantes pour décider de l'appartenance d'un mot à une classe d'entités nommées particulière. Il est primordial que les caractéristiques mises en place captent ces particularités de façon efficace, sans générer de bruit, pour éviter un

[124] La bibliothèque de code CRF utilise des adaptateurs pour accéder aux données, avec un mécanisme d'itération. Concrètement, un adaptateur est une classe qui sert d'interface entre un format de données spécifique (le nôtre, par exemple), et un format connu et manipulable par les classes natives de la bibliothèque. Dans notre cas, les séquences de données correspondent à des dépêches de presse annotées et les itérateurs permettent de passer d'un fichier de dépêche à l'autre.

surapprentissage (ou *over-fitting*)[125]. C'est à ce stade qu'une expertise linguistique est importante dans le processus d'apprentissage automatique.

L'expérience acquise lors de l'évaluation du Stanford NER, ainsi que l'étude de la littérature, nous ont permis de définir une liste de caractéristiques que nous décrivons ici. Nous détaillons leurs intérêts et inconvénients et concluons par des perspectives d'évolution.

(1) Mots

La caractéristique la plus basique à mettre en œuvre consiste simplement à observer le lien qui existe entre une forme de surface (le mot lui-même) et son étiquette (ici, sa classification en tant qu'entité nommée). L'algorithme d'apprentissage mémorise les couples (mot, étiquette) rencontrés, permettant de distinguer toutes les caractéristiques de ce type, mais permettant également de retrouver cette caractéristique lorsqu'elle se reproduit :

- Lors de l'entraînement : si elle se reproduit souvent avec une classe d'entité nommée, c'est qu'elle est très caractéristique de ce type d'entité nommée.
- Lors de la phase d'annotation : si un mot possède cette caractéristique, il est fort probable qu'il soit de la même classe d'entité nommée que les mots du corpus d'entraînement qui la possédait.

Cette fonction permet donc d'identifier de nombreux mots ou groupes de mots correspondant à des entités nommées rencontrés dans le corpus d'entraînement. Ainsi, en utilisant uniquement cette caractéristique, on obtient un F-score légèrement supérieur à 60 %.

Toutefois, ces caractéristiques restent limitées. Utilisées seules, elles ne permettent de détecter que les mots déjà rencontrés ; il est également impossible d'utiliser ce type de caractéristiques pour des tâches de désambiguïsation. De plus, on va créer une caractéristique unique pour chaque forme des mots rencontrés, sans être capable d'effectuer des regroupements vers la forme de base d'un mot (prise en compte des variations morphologiques, formes de verbes, genre et nombre d'un nom commun ou d'un adjectif...).

(2) Mot précédent et mot suivant

Le contexte autour du mot courant est une caractéristique intéressante pour détecter des entités nommées. Commençons par examiner le mot juste avant celui que l'on souhaite annoter. Cette démarche peut être comprise comme la recherche de mots déclencheurs. En effet, pour introduire un lieu, on peut supposer qu'en français on trouvera souvent des prépositions telles que « à », « dans », « vers»… alors qu'on trouvera plus souvent des titres tels que « Madame », « Monsieur », « Ministre », « Professeur »… pour introduire une entité de type personne. De même, on introduit une caractéristique correspondant au mot suivant celui qu'on souhaite annoter.

(3) Casse du mot

Les entités nommées sont souvent des noms propres et donc des mots commençant par une majuscule (nom de personne ou de lieu…) ; il est donc utile de s'intéresser à la casse du mot pour aider à la détection d'entités nommées.

[125] Phénomène qui se produit lorsqu'un modèle statistique décrit des propriétés qui s'avèrent trop spécifiques aux exemples d'entraînement ; l'apprentissage risque alors d'être bruité et de contenir des erreurs.

Pour effectuer cette détection, on compare les mots rencontrés à des expressions régulières[126] :

- $[A-Z][a-z]+$: vérifie qu'un mot commence par une majuscule.
- $[a-z]+$: vérifie qu'un mot est entièrement en minuscules ; en effet, il est aussi important de créer des caractéristiques qui vont indiquer qu'un mot *n'est pas* une entité nommée.
- $[0-9]+$: vérifie qu'un « mot » est uniquement composé de chiffres.
- $[A-Z]+$: vérifie qu'un mot est entièrement en majuscules ; c'est utile pour les sigles ; notons que dans les dépêches de presse, le premier mot est entièrement en majuscules et correspond au lieu sur lequel porte la dépêche.

On cherche également à détecter des patrons un peu moins triviaux, par exemple des mots contenant des majuscules à d'autres emplacements qu'au début du mot, quand ils ne sont pas entièrement en majuscules. C'est un cas intéressant lorsqu'on cherche à détecter des produits technologiques, des noms composés ou encore des noms d'entreprises.

- $[A-Z][a-z]+[A-Z][a-z]*$: McCallum, Bluetooth…
- $[a-z]+[A-Z][A-Za-z]*$: iPhone, eeePC…

Notons que les caractéristiques portant sur la forme des mots ne peuvent pas être utilisées seules : si l'on va aisément détecter qu'un mot commençant par une majuscule est une entité nommée, sa seule casse ne permettra pas d'en déterminer la classe. Ces caractéristiques permettent d'augmenter le rappel, lorsqu'elles sont utilisées conjointement avec d'autres types de caractéristiques.

(4) Transitions

Tout comme les enchaînements entre les mots peuvent se révéler intéressants, les dépendances sur les suites d'étiquettes peuvent aussi apporter des informations utiles à la détermination des classes d'entités nommées. Pour illustrer ce point sur un exemple concret, on imagine facilement que deux mots consécutifs puissent constituer une entité type « Personne » (dans le cas où prénom et nom se suivent). En revanche, il est très rare de retrouver une suite de mots où deux entités (de type « Personne Technologie » ou « Lieu Personne » par exemple) se suivent de façon contigüe.

(5) Lexémisation

Les différentes formes fléchies d'un mot partagent en principe[127] la même racine : « aimer », « aime », « aimé », « aimions » ou « aimât » ont la même racine *aim-*. La lexémisation permet donc d'effectuer des regroupements de mots provenant d'une même racine. Nous avons utilisé une lexémisation en amont de la génération des caractéristiques décrites ici en (1) et (2), ce qui améliore la reconnaissance des entités nommées et réduit la taille des ensembles de caractéristiques.

(6) Préfixes et suffixes

Les préfixes et les suffixes des mots entourant le mot considéré peuvent donner des informations d'ordre morphosyntaxique. Par exemple, en anglais ou en français, les deux ou trois dernières lettres des verbes sont un bon indicateur du temps, du mode, du genre et du nombre. L'observation des préfixes permet une « lexémisation du pauvre ». Cette caractéristique a pour paramètre la longueur des préfixes ou suffixes à observer.

---

[126] Pour éviter que la détection de ces différentes expressions régulières se révèle coûteuse en temps machine, elles sont précompilées.

[127] Cf. la discussion en section II.A.3.d).

(7) Gazettes

La détection d'entités nommées peut être facilitée par l'ajout de caractéristiques qui vont tester l'appartenance du mot à une liste prédéfinie. Nous avons capitalisé sur le mécanisme de gazettes contextuelles décrites en section V.B.2, page 95.

Prenons l'exemple d'une liste de lieux. Lors de l'entraînement, on trouvera régulièrement des mots appartenant à cette liste étiquetés comme étant des lieux. Lors de la phase d'annotation, la probabilité pour qu'il s'agisse d'un lieu augmentera pour un mot appartenant à cette liste, même s'il n'apparaît jamais dans le corpus d'entraînement. Ce type de caractéristique permet de modifier les listes sans avoir à entraîner à nouveau le CRF.

(8) Etiquetage morphosyntaxique

L'objectif est ici d'utiliser comme caractéristique la partie du discours des mots du corpus d'entraînement et du corpus où détecter les entités nommées. On peut de cette façon distinguer les homographes.

d) *Evaluation sur un corpus en anglais*

Nous avons évalué l'évolution du temps d'entraînement en fonction de la taille du corpus d'entraînement en fixant le nombre de classes et les options choisies, puis en effectuant l'entraînement sur plusieurs corpus contenant un nombre croissant de mots.

Nous avons mesuré les performances sur un corpus anglais avec trois classes d'entités nommées (personnes, lieux géographiques, organisations), la taille de corpus variant entre 1 000 et 300 000 mots. Ces tests permettent de mettre en évidence l'augmentation du temps d'entraînement en fonction du nombre de mots[128]. La charge mémoire reste réduite lorsque le nombre de classes d'entités nommées est faible. Elle augmente avec le nombre de classes d'entités nommées à reconnaître. Cette caractéristique a rendu, en pratique, quasi impossibles les tests portant sur plus de cinq classes d'entités nommées sur une machine de test disposant de 2 Go de mémoire vive.

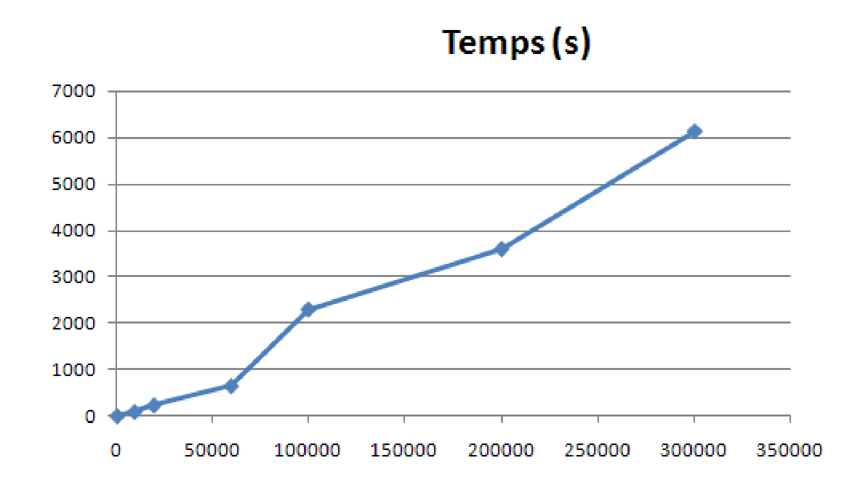


Figure 25 : Temps d'apprentissage sur le corpus anglais d'entités nommées

[128] Précisons que depuis l'implémentation initiale, l'équipe Proxem a effectué un travail de fond d'optimisation des performances ; avec la version la plus récente, le temps d'apprentissage est devenu sensiblement plus court qu'avec les autres implémentations que nous avons pu évaluer (CRF++ et Wapiti).

Nous avons évalué notre implémentation (précision, rappel, et F-mesure) en comparant ses annotations avec celles obtenues par OpenCalais (automatiquement, puis en partie corrigées humainement). La présence simultanée des annotations attendues et des annotations obtenues permet de compter les vrais positifs, les vrais négatifs et les faux positifs.

La figure 26 montre le gain de F-mesure observé lorsqu'on augmente le nombre de mots dans le corpus d'entraînement. On constate que l'amélioration du F-score est lente, mais réelle, par rapport à la taille du corpus et donc au temps d'entraînement.

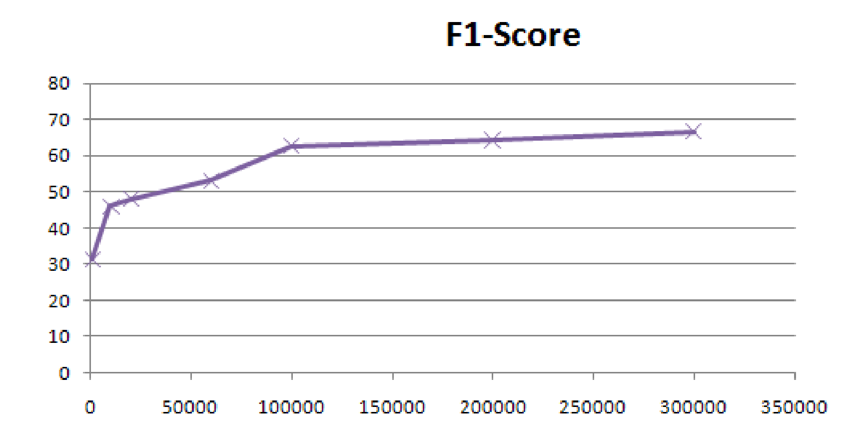


Figure 26 : F-score sur le corpus anglais en fonction de la taille du corpus d'apprentissage

Entre 100 000 et 300 000 mots, on gagne quasiment 4 points de F-score tout en multipliant le temps d'apprentissage environ par 3 (on est passé de 38 minutes à 93 minutes). Une telle progression de F-score reste très significative, et montre l'intérêt, malgré le temps d'entraînement accru, d'effectuer l'apprentissage sur un corpus important.

e) Evaluation sur un corpus en français

Les tests effectués sur le corpus de dépêches de presses en français ont permis d'obtenir des résultats très similaires à ceux obtenus pour l'anglais, comme le montrent la figure 27 et la figure 28.

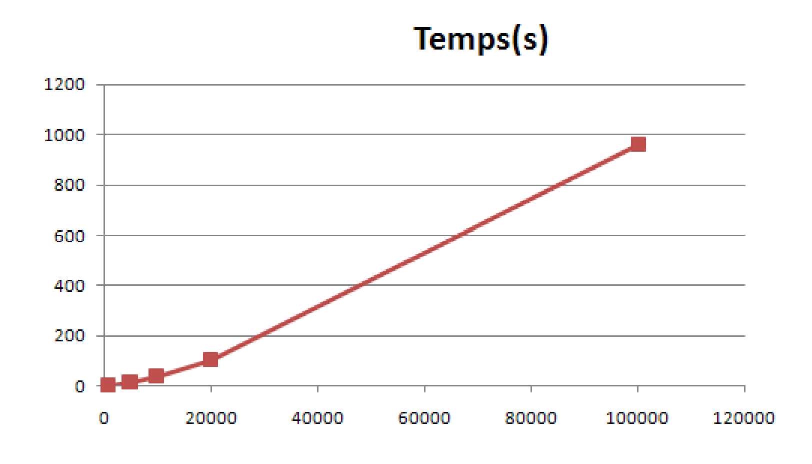


Figure 27 : Temps d'apprentissage sur le corpus français d'entités nommées

F1-score

80
70
60
50
40
30
20
10
0

0 20000 40000 60000 80000 100000 120000

Figure 28 : F-score sur le corpus français en fonction de la taille du corpus d'apprentissage

f) *Interface graphique*

Nous avons développé une interface graphique permettant, pour la tâche à accomplir (entraînement, annotation, test des performances), d'effectuer simplement un choix des caractéristiques à utiliser, sans avoir à modifier le code source. Au lancement du programme, l'initialisation du modèle de CRF crée tous les objets correspondant aux générateurs de caractéristiques, grâce à cette liste et aux paramètres choisis par l'utilisateur.

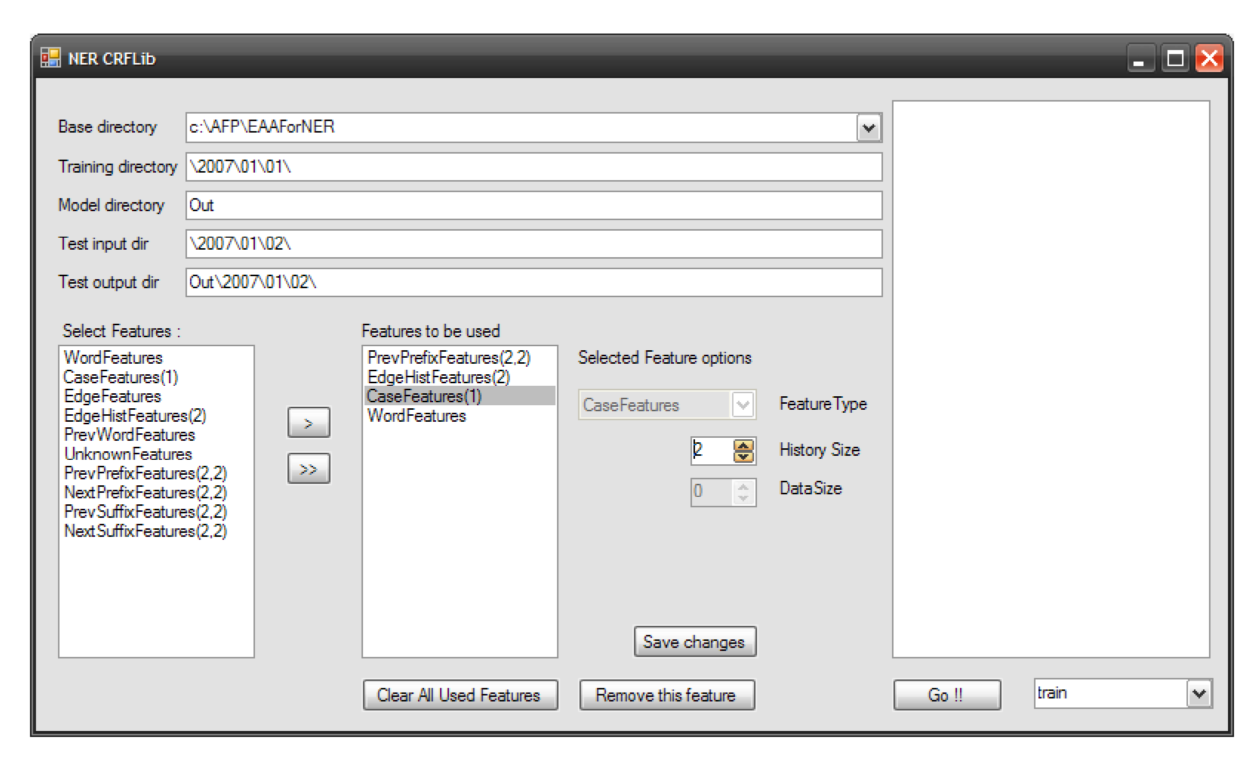


Figure 29 : Interface graphique de l'outil d'apprentissage

g) *Résultats*

L'implémentation de notre détecteur d'entités nommées utilise une bibliothèque de code CRF et permet de définir des caractéristiques spécifiques à un projet (venant éventuellement compléter celles décrites plus haut en sous-section c). Les caractéristiques mises en place permettent l'obtention de F-scores cohérents par rapport à ceux obtenus avec le Stanford NER.

Notre première mise en œuvre effective a concerné le projet SCRIBO (Cf. chapitre VI.A, page 132). Il s'agissait de détecter les personnes, lieux, organisations et montants monétaires cités dans des articles de presse en français. Nous avons manqué de temps sur ce projet pour améliorer les caractéristiques prises en compte pendant l'apprentissage. Les résultats que nous avons obtenus figurent dans le tableau 11 ci-dessous ; on y constate que les organisations semblent plus difficiles à identifier que les personnes ou les lieux, notamment en ce qui concerne le rappel.

Classe d'entité	précision	rappel	F-mesure
Personne	0,8515	0,8123	0,8314
Lieu	0,8882	0,8886	0,8881
Organisation	0,7266	0,4909	0,5852
Monnaie	1	0,9626	0,9809

Tableau 11 : Résultats de la reconnaissance d'entités nommées sur le projet SCRIBO

Sur un autre projet, portant sur l'analyse d'avis de consommateurs (Cf. chapitre VI.D, page 141), nous avons cherché à détecter des classes d'entités différentes : des produits, des marques, des enseignes concurrentes et des concepts (tels que le risque juridique, le risque sanitaire ou le régime sans gluten, par exemple). Nous avons introduit plusieurs autres caractéristiques d'apprentissage, notamment en prenant en compte les dépendances syntaxiques. Nous avons aussi amélioré le mécanisme standard en ajoutant un traitement particulier pour reconnaître les produits composés de la forme NP PP à partir des produits déjà reconnus[129] (Cf. page 143, section VI.D.3.c). Avec un corpus d'apprentissage constitué d'environ 400 000 avis annotés initialement avec le système de reconnaissance d'entités utilisant des gazettes contextuelles, nous avons obtenu une F-mesure égale à 0,971 (avec une précision de 0,951 et un rappel de 0,992).

Précisons enfin que nous n'avons pas encore procédé à d'évaluation sur un corpus « standard » (tel qu'ESTER par exemple), mais que nous prévoyons de le faire dans le futur.

6. Conclusion

Les performances de la reconnaissance d'entités nommées par CRF varient en fonction du volume du corpus d'apprentissage et de la qualité de ses annotations. Au-delà du travail d'ingénierie et d'implémentation d'algorithmes que cela représente dans la plate-forme, nous considérons cette tâche comme un module essentiel dans l'interface sémantique-syntaxe car elle représente à ce jour la meilleure approche pour effectuer une désambiguïsation lexicale fine[130] et contribuer au calcul de la RSém. Nous verrons en partie VII comment nous combinons ce module avec d'autres pour constituer notre interface syntaxe-sémantique.

[129] Par exemple, « chocolat noir aux noisettes » ou « canard à l'orange ».

[130] Certes sur un nombre restreint de termes, qui correspondent aux classes d'entités à reconnaître dans le corpus considéré.

C. Extraction de relations

1. Introduction

a) Des rôles thématiques aux rôles sémantiques

Nous avons vu dans le chapitre précédent comment reconnaître des entités nommées, de type SOCIETE ou PERSONNE, par exemple. Nous allons à présent nous attacher à extraire des relations entre les unités sémantiques que sont les unités lexicales et les entités nommées. Chaque unité sémantique est traitée comme un prédicat logique avec un certain nombre d'arguments. Les relations sémantiques sont donc des relations prédicat-argument.

Plusieurs niveaux de finesse sont possibles dans une telle opération, en fonction des objectifs visés. L'approche la plus ancienne, héritée des pratiques de la logique des prédicats, consiste à différencier les arguments simplement en les numérotant : arg1, arg2, arg3… (ou arg0, arg1, arg2…) en suivant en général un ordre d'oblicité croissante (sujet < objet direct < objet indirect < complément oblique). Le rôle joué par chaque argument d'une unité lexicale est décrit dans l'entrée lexicale de celle-ci[131]. Il n'y aucune généralisation faite sur le lexique : un arg2 est simplement le deuxième argument d'une unité lexicale et ne présuppose pas un rôle particulier. Seule la consultation du lexique permet de savoir à quoi il renvoie exactement.

Une opération plus riche consiste à non seulement différencier les arguments, mais aussi à les typer et à les nommer. Nous distinguerons dans la suite deux niveaux de typage des rôles : l'un, très général, s'applique à des classes de verbes (rôle thématique) ; l'autre est spécifique à un prédicat spécifique ou une relation particulière (rôle sémantique).

Les rôles thématiques (Cf. page 57 la section IV.B.3.b) sont un ensemble fini de types de participants, utilisés pour décrire les comportements des verbes, indépendamment de leur construction syntaxique. Ils caractérisent, du moins d'une façon superficielle, la relation qu'un verbe et ses arguments entretiennent. Par exemple, dans une phrase comme « *Microsoft rachète Powerset* », Microsoft joue le rôle d'*Agent* et Powerset celui de *Patient*. Nous présenterons notre approche pour calculer les rôles thématiques en section 3 (page 110).

Quand on souhaite obtenir un étiquetage plus précis, on peut spécifier plus finement la sémantique de ces rôles. Dans l'exemple précédent, Microsoft est l'*Acheteur* et Powerset la *Société achetée* ou d'une façon plus générale la *Marchandise échangée*[132]. Nous verrons en section 4 (page 111) comment nous procédons à ce calcul.

On peut alors mettre en relation les unités lexicales qui possèdent les mêmes rôles sémantiques et qui appartiennent au même Frame au sens de FrameNet (Cf. IV.D.3 page 83). ACHETER et VENDRE qui possèdent les mêmes rôles sémantiques (*Acheteur*, *Vendeur*, *Marchandise*, *Montant*) sont potentiellement des conversifs. Rôles sémantiques et rôles thématiques sont des notions distinctes. Ainsi l'*Acheteur* est l'*Agent* de ACHETER mais le *Destinataire* de VENDRE, tandis que l'*Agent* de VENDRE est le *Vendeur*.

---

131 Cf. par exemple (Mel'čuk, 1988a).

132 D'autres entités sont intéressantes à détecter dans un tel contexte, comme le *Montant* de l'acquisition, si cette information apparait dans le texte.

Nos composants d'étiquetage de rôles travaillent sur la sortie d'un analyseur syntaxique en dépendances. Si les deux types de calculs utilisent des ressources lexico-sémantiques très différentes, le mécanisme de calcul utilisé dans les deux cas est le même : la recherche de sous-graphe dans le graphe de dépendances, en utilisant plusieurs patrons de recherche ; nous abordons ces points dans les deux sous-sections suivantes.

Ce mécanisme peut être utilisé sans changement d'algorithme aussi bien sur la RSyntS que la RSyntP. Dans ce dernier cas, le rappel est en principe amélioré. Nous présenterons notre approche de l'analyse syntaxique profonde en section 2 (page 108).

b) Recherche de sous-graphe dans un graphe

Nos composants d'étiquetage de rôles thématiques ou sémantiques sont implémentés d'une façon similaire. Leur algorithme consiste essentiellement à chercher un sous-graphe dans un graphe, en vérifiant éventuellement des contraintes de sélection. Le sous-graphe correspond au patron morphosyntaxique de la relation que nous souhaitons reconnaître au sein du graphe ; par exemple, la figure 30 correspond à une relation d'acquisition d'une société (*SociétéAchetée* dans la figure) par une autre (*Acheteur* dans la figure). La figure 31 illustre l'analyse syntaxique en dépendances d'une phrase, qui contient (en gras) le sous-graphe correspondant à la relation d'acquisition.

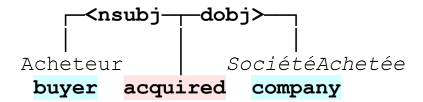


Figure 30 : Patron morphosyntaxique de la relation d'acquisition d'une société par une autre

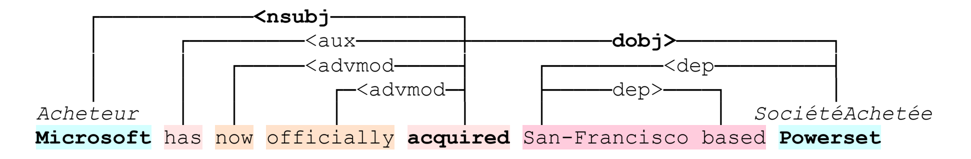


Figure 31 : Analyse en dépendances d'une phrase où on reconnaît une acquisition

Notons que, dans l'état actuel de l'implémentation, nous ne cherchons pas à extraire directement des relations à un niveau interphrastique. Nous pourrions, en revanche, utiliser les composants de résolution d'anaphores à cet effet.

L'implémentation actuelle de la recherche de sous-graphe dans un graphe est basée sur la génération de code PROLOG[133]. Dans une étape préparatoire, nous traduisons la construction à reconnaître[134] dans le programme PROLOG équivalent. Lors de l'analyse effective d'un texte, ce programme s'applique à un autre programme PROLOG, décrivant le graphe syntaxique d'une analyse en dépendances de chaque phrase. Le mécanisme d'unification permet alors de trouver –ou non– la construction cherchée, précompilée lors de l'étape préparatoire.

[133] PROLOG (PROgrammation LOGique) est un langage déclaratif utilisant le mécanisme d'unification avec retour arrière. Il est souvent utilisé dans des applications d'intelligence artificielle. Un programme PROLOG se constitue de prédicats décrivant des faits ou des règles. On utilise un tel programme pour chercher si un but donné est atteint ou non. Ce langage est donc bien adapté pour chercher un sous-graphe (par exemple la réalisation d'un cadre de sous-catégorisation) à l'intérieur d'un graphe (correspondant à la sortie de l'analyse syntaxique en dépendances d'une phrase).

[134] Par exemple, la description VerbNet en XML des cadres de sous-catégorisation d'une classe de verbes.

Ce mécanisme donne satisfaction mais est relativement gourmand en puissance de calcul. Nous envisageons deux voies pour en améliorer les performances. Une possibilité serait de remplacer PROLOG par une boîte à outil spécialisée dans la réécriture de graphes, GrGen, qui a été optimisée pour rechercher rapidement des sous-graphes. Une autre approche, qui permettrait de traiter efficacement des corpus de grande taille, consisterait à mettre en œuvre une base de données au format RDF ; l'analyse syntaxique en dépendances du corpus y est alors stockée sous forme de triplets ; la recherche de sous-graphe revient alors à émettre une simple requête SPARQL (Cf. page 192). Nos tests préliminaires montrent que l'utilisation de GrGen donne une amélioration des performances par rapport au mécanisme PROLOG ; nous n'avons pas encore pu les comparer à celles d'un moteur SPARQL.

Pour finir, précisons que l'opération de recherche de sous-graphe est réalisée en testant les éventuelles contraintes de sélection. Ce test est effectué, en fonction du contexte, soit en cherchant de telles contraintes dans notre lexique sémantique, soit après une étape de reconnaissance d'entités nommées. On voit donc qu'en fonction des objectifs visés, la reconnaissance de sous-graphe dans un graphe peut être utilisée soit comme mécanisme de vérification de contraintes, soit comme mécanisme d'inférence.

c) Un obstacle : la multiplicité des paraphrases

La richesse paraphrastique du langage permet d'exprimer une relation comme « *SOCIETE1 rachète SOCIETE2* » de nombreuses façons différentes. Pour éviter de multiplier à l'infini les patrons morphosyntaxiques correspondant aux réalisations possibles, nous distinguons différents niveaux possibles d'expression des paraphrases.

Un premier niveau de complexité provient de l'utilisation d'une construction syntaxique différente de la forme canonique « sujet verbe complément ». On peut par exemple changer la diathèse d'un verbe (« *Powerset a été rachetée par Microsoft* ») ou utiliser une proposition relative (« *Microsoft, qui a racheté Powerset, …* ») ; la combinaison des deux est aussi possible (« *Powerset, qui a été rachetée par Microsoft, …* »). Même si une combinatoire de ce type de constructions existe, leur nombre de variantes semble fini. Pour prendre en compte ce type de paraphrases, nous effectuons la recherche de sous-graphe non sur la RSyntS, mais sur la RSyntP ; cette approche permet d'améliorer le rappel du système. L'implémentation de l'interface RSyntS ⇒ RSyntP est présentée en section 2.

Des variantes plus complexes peuvent être formulées en utilisant un synonyme du verbe prédicat (« *Microsoft acquiert Powerset* ») ou une construction avec une nominalisation (« *rachat de Powerset par Microsoft* », « *Microsoft procède à l'acquisition de Powerset* »). Une expression sémantiquement équivalente ou une implicature peuvent aussi être utilisées (« *Microsoft prend le contrôle de Powerset* »). Dans notre implémentation actuelle, ces variantes doivent être explicitées ; toutefois, rien n'empêche d'utiliser le mécanisme d'apprentissage de paraphrases présenté en section IV.C.3 (page 68) pour amorcer une liste de variantes.

2. Analyse syntaxique profonde

Nous allons à présent illustrer concrètement la transition réalisée par l'interface RSyntS ⇒ RSyntP, c'est-à-dire allant de la syntaxe de surface vers la syntaxe profonde (Cf. la section II.B.1, page 14).

La phrase que nous utiliserons pour illustrer cette transition est « *the general to whom Lincoln gave all powers in Washington captured Lee's troops during the Battle of Gettysburg* » (« le général à qui Lincoln a donné tous les pouvoirs à Washington a capturé les troupes de Lee pendant la bataille de Gettysburg ») ; son analyse syntaxique de surface[135] est représentée figure 32.

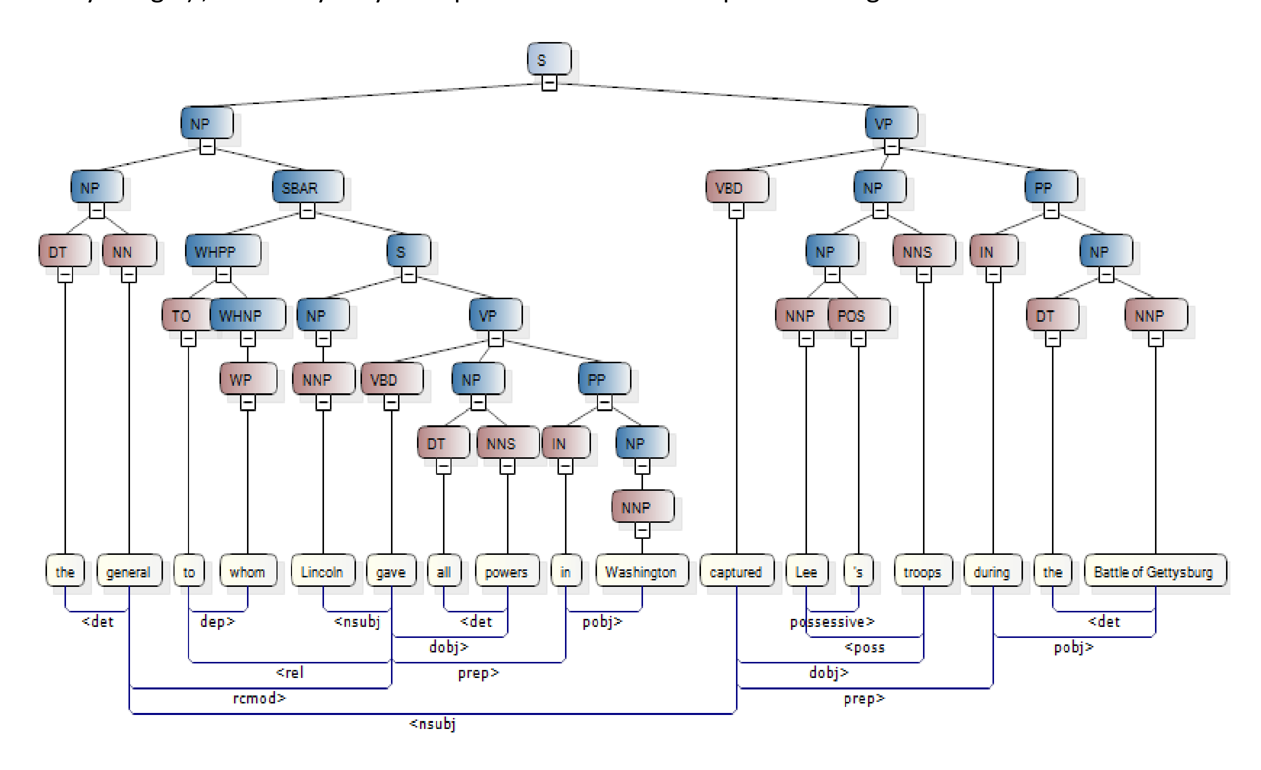


Figure 32 : Représentation syntaxique de surface d'une phrase en anglais

a) Calcul des dépendances syntaxiques profondes

Pour calculer les dépendances en syntaxe profonde, nous partons d'une copie de l'arbre de dépendances en syntaxe de surface. Nous y appliquons ensuite des restructurations successives, en cherchant à y reconnaître une liste finie de phénomènes linguistiques, correspondant aux formes verbales passives, aux relatives, aux subordonnées… Nous appliquons des règles de réécriture pour modifier, créer ou supprimer des dépendances. Les paires de dépendances introduisant des groupes prépositionnels (verbe vers préposition, préposition vers tête du groupe nominal) sont fusionnées pour que le verbe pointe directement vers la tête nominale du groupe prépositionnel, en mémorisant la préposition régime.

La figure 33 montre l'analyse de la phrase d'exemple avec les dépendances en syntaxe de surface (au-dessus des mots) et en syntaxe profonde (en-dessous des mots). On remarquera la dépendance syntaxique profonde `PropObject(to)` qui identifie « *general* » en tant que complément d'objet indirect du verbe « *give* ».

[135] On remarquera toutefois que l'expression multi-mots « Battle of Gettysburg » y est déjà reconnue.

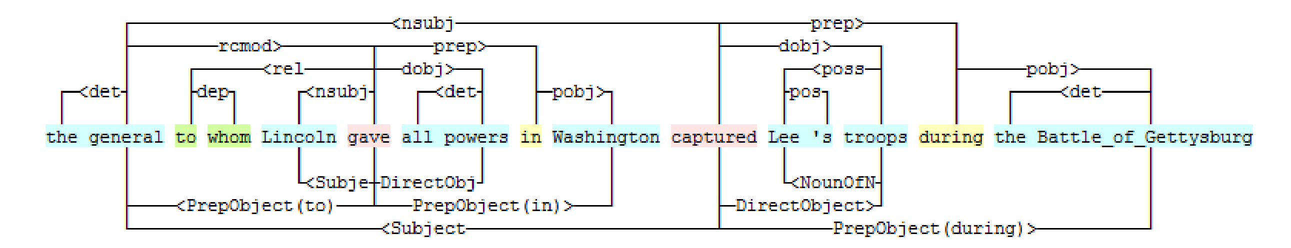


Figure 33 : Syntaxe de surface (au-dessus des mots) et syntaxe profonde (en dessous)

b) *Identification des compléments de temps et d'espace*

Nous avons aussi implémenté une détection basique des compléments de temps et d'espace. Un système de règles de réécriture est utilisé pour identifier les dépendances syntaxiques profondes correspondantes. Par exemple, si une dépendance de type `PrepObject` utilise une préposition compatible avec un complément de temps et que le groupe prépositionnel est une date ou un nom qui a pour hyperonyme un synset comme {EVENT}, {DYNASTY} ou {DECADE}, alors la dépendance est affinée en `TimeComplement`, comme illustré figure 34. De même, si une dépendance de type `PrepObject` utilise une préposition compatible avec un complément de lieu et que le groupe prépositionnel est un nom qui a pour hyperonyme un synset comme {LOCATION}, {REGION}, {STRUCTURE}, {LAND}, {BODYOFWATER} ou {GEOLOGICALFORMATION}, alors la dépendance devient `SpaceComplement`.

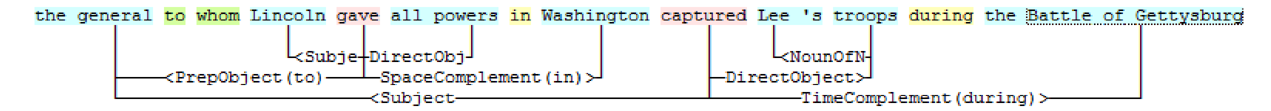


Figure 34 : Extraction des compléments de temps et de lieu

À ce stade, il est aisé d'extraire la structure actancielle prédicat-arguments en regroupant l'ensemble des dépendances en syntaxe profonde gouvernées par un même verbe ou un même nom prédicatif. La phrase d'exemple précédente donne deux prédications :

- Gave(**Subject**: Lincoln, **DirectObject**: powers, **PrepObject**: *to* general, **SpaceComplement**: *in* Washington).
- Captured(**Subject**: general, **DirectObject**: troops, **TimeComplement**: *during* Battle_of_Gettysburg).

3. Etiquetage des rôles thématiques

(Chaumartin, 2006) décrit en détail notre implémentation de l'étiquetage des rôles thématiques. Plusieurs ressources du lexique sémantique (WordNet, VerbNet et SUMO) sont mises à contribution. La syntaxe des classes de verbes décrite dans VerbNet est utilisée (avec le processus décrit en V.C.1.b) pour reconnaître les cadres de sous-catégorisation des verbes des phrases fournies en entrée. Le graphe des noms de WordNet, les gloses désambiguïsées des synsets et l'ontologie SUMO servent à la vérification des contraintes de sélection.

Chaque prédicat calculé lors de l'analyse syntaxique profonde de l'exemple est alors étiqueté. On notera que certains rôles thématiques optionnellement prévus dans le cadre de sous-catégorisation

du verbe peuvent ne pas être explicitement renseignés (ici, *Asset* et *Source* dans le premier prédicat, *Source* et *Beneficiary* dans le second).

– E1: gave(**Agent**: Lincoln <animate>, **Theme**: powers, **Recipient**: general <animate>, Asset: ?, Source: ?, **Location**: *in* Washington).

– E2: captured(**Agent**: general <animate>, **Theme**: troops, Source: ?, Beneficiary: ?, **Time**: *during* Battle_of_Gettysburg).

L'étiquetage des rôles thématiques contribue directement à la désambiguïsation lexicale. En effet, dans le cas de l'exemple, VerbNet restreint les sens possibles du verbe et de ses actants dans WordNet par application des contraintes de sélection :

- Seuls huit sens du verbe GIVE, parmi les quarante-neuf énumérés par WordNet, sont compatibles avec l'analyse syntaxique de surface de la phrase.
- Le nom GENERAL étant contraint par un trait <animé>, ses deux sens possibles sont GENERAL#1 et #2 ; le troisième sens décrit dans WordNet (général par opposition à particulier) est exclu.
- Le seul sens possible du verbe CAPTURE dans le contexte est CAPTURE#5.

VerbNet décrit aussi la sémantique fine de chaque cadre de sous-catégorisation avec un jeu de deux cents prédicats de base. Par exemple, la traduction du premier prédicat dans ces concepts élémentaires donne :

- Has_possession(start(E1), Agent=Lincoln, Theme=powers).
- Has_possession(end(E1), Recipient=general, Theme=powers).
- Transfer(during(E1), Theme=powers).
- Cause(Agent=Lincoln, E1).

Notre implémentation de l'étiquetage des rôles thématiques nécessite des développements complémentaires pour être réellement utilisable sur des textes tout-venant. Aujourd'hui, nous considérons qu'elle n'identifie correctement les rôles que sur un tiers des textes. Cette limitation nous semble liée principalement à deux facteurs : d'une part, la couverture perfectible de la ressource VerbNet, et d'autre part la complexité intrinsèque de la tâche. Notons que pour l'instant, l'étiquetage des rôles thématiques dans Antelope ne fonctionne que pour l'anglais, car il dépend de VerbNet.

4. Etiquetage de rôles sémantiques

a) Objectifs

Pour calculer un étiquetage plus précis qu'un rôle thématique, Antelope intègre un composant d'extraction d'information, orienté vers la tâche de remplissage de patrons (ou *template filling* en anglais). Son but est de détecter des prédicats dans un texte et de remplir automatiquement les valeurs des arguments de ces prédicats. Notre objectif est triple : nous cherchons d'abord à privilégier des réponses précises ; ensuite, à exprimer aussi simplement que possible les informations cherchées ; enfin, à être indépendant d'une langue donnée ou d'un analyseur particulier. Le composant utilisant la sortie d'un analyseur syntaxique intégré à Antelope, l'extraction d'information fonctionne sur des textes français ou anglais.

b) Architecture technique du composant d'extraction d'information

La figure 35 détaille les classes utilisées par le composant. Un prédicat (classe `Template`) est décrit par des exemples (classe `Sample`) qui sont des paraphrases d'une construction donnée. Ces exemples peuvent être formulés dans l'une des langues pour lesquelles Antelope dispose d'un analyseur syntaxique (actuellement le français ou l'anglais).

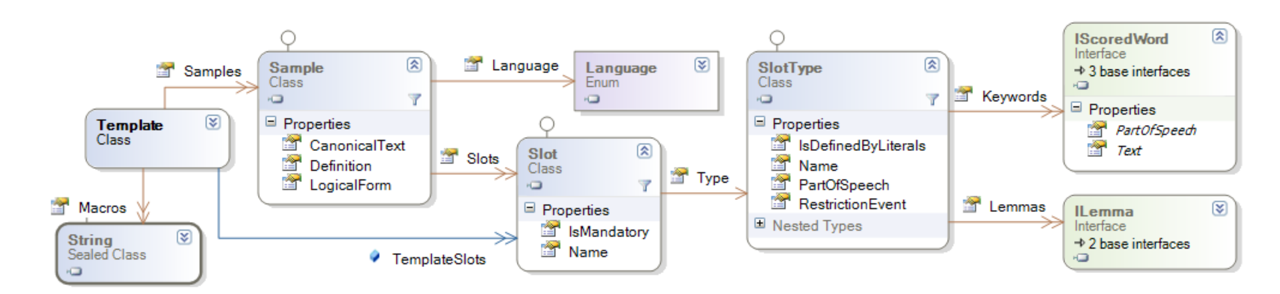


Figure 35 : Diagramme des classes utilisées par le composant d'extraction d'information

Une relation a un ou plusieurs arguments (classe `Slot`) d'un certain type (classe `SlotType`). Un argument peut être obligatoire ou optionnel. Par exemple, dans la relation `acquisition` (de société par une autre), les deux arguments *Acheteur* et *SociétéAchetée* sont obligatoires ; un troisième argument *Montant* n'est pas toujours présent et peut donc être considéré comme optionnel. Un type d'argument peut être associé à un ensemble de mots-clés ou de lemmes.

(1) Typage des arguments

Chaque argument a un type défini (classe `SlotType`). A minima, ce type est une partie du discours (nom, verbe…). Des contraintes plus précises peuvent être exprimées de quatre façons pour appliquer des contraintes de sélection sur les arguments d'un prédicat.

Un type d'argument peut être défini d'une façon **extensionnelle** par une liste de vocables. C'est utile pour énumérer un ensemble fini, comme les pays d'Europe, des secteurs de l'industrie, etc.

Un type d'argument peut aussi être défini d'une façon **intensionnelle**, grâce au lexique sémantique, avec une liste d'hyperonymes qui servent de point de départ. La hiérarchie de WordNet (pour les noms ou verbes) est alors utilisée pour ajouter récursivement leurs hyponymes à la liste. Par exemple un argument *EndroitOùManger* est défini initialement avec les lemmes RESTAURANT et BAR ; après une phase de recherche récursive des hyponymes, *EndroitOùManger* est enrichi de termes tels que BISTRO, STEAKHOUSE et CAFETERIA. Le système considère par défaut le premier sens du mot dans WordNet (son sens le plus fréquent). Si besoin, l'utilisateur peut préciser un sens particulier (par exemple CANTEEN#2). L'utilisateur du composant peut aussi imposer un seuil minimal de fréquence d'apparition des lemmes pour éviter d'ajouter à la liste des termes trop rares lors de la recherche récursive des hyponymes.

Un type d'argument peut également être vérifié à l'aide d'une **expression régulière**. Cela s'applique bien aux éléments qui ont une forme particulière (numéro de téléphone, numéro ISBN…). Nous définissons par exemple de cette manière un type *NomPropre* qui correspond simplement aux noms qui commencent par une majuscule.

Enfin, le type d'un argument peut être vérifié **dynamiquement** à l'exécution d'une façon plus souple qu'en cherchant des termes dans un lexique. Cela permet d'effectuer un test *ad hoc* pour vérifier les contraintes de sélection en tenant compte du contexte. Ce type de vérification est intéressant pour des éléments qui varient dans le temps ou dans l'espace (les spectacles à l'affiche dans un lieu donné, par exemple), ou qui dépendent d'un utilisateur particulier d'une application (contacts de son carnet d'adresse, rendez-vous de son agenda…).

(2) Extraction des arguments

Chaque paraphrase est transformée en une forme logique, qui teste si une phrase donnée correspond au patron attendu, puis en extrait la valeur des arguments. Le mécanisme utilisé est la recherche de sous-graphe au sein d'un graphe (Cf. V.C.1.b). Une fois le sous-graphe trouvé, l'algorithme connaît les mots reliés entre eux par les dépendances exprimées dans le patron morphosyntaxique. Il reste à tester la partie du discours de chaque mot et les éventuelles contraintes de sélection. L'ordre des mots (consécutifs, mais non forcément contigus) est également vérifié.

c) Evaluation sur un exemple d'acquisition de sociétés

La relation `acquisition(`*`Acheteur`*`,` *`SociétéAchetée`*`)` est associée, comme montré en figure 36, à onze réalisations linguistiques en anglais[136,137]. Le composant interroge d'abord des moteurs de recherche[138] avec les mots clés de chaque paraphrase et collecte une liste de documents, qui sont segmentés en phrases. Celles contenant tous les mots clés (dans le même ordre que celui exprimé dans le patron morphosyntaxique) sont retenues en tant que phrases candidates, puis testées par le composant d'extraction d'information.

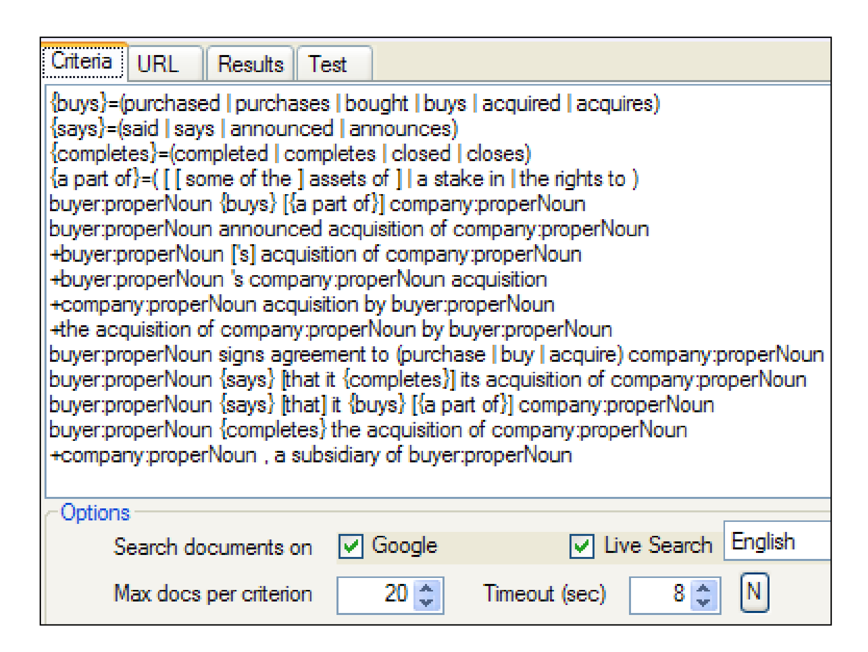


Figure 36 : Interfaces de saisie des critères de recherche

Nous avons évalué le composant en le testant sur les rachats de sociétés et prises de participations effectués par Microsoft. L'application trouve 2 160 documents à partir du Web, contenant 4 367

[136] Ces différentes paraphrases ont été créées manuellement par examen d'un corpus de dépêches financières.
[137] Les mots entre accolades sont des macros qui seront substituées ; leur intérêt est d'alléger l'expression des paraphrases en définissant une fois pour toutes des motifs fréquemment répétés. Le préfixe + indique un GN.
[138] Disposant d'une API, comme Yahoo! et Microsoft Bing (celle de Google a été suspendue en novembre 2010).

phrases candidates, et extrait une information à partir de 1 353 phrases (en un peu moins d'une heure de traitement). À peu près 10 % de ces résultats sont erronés du fait d'une segmentation ou d'une analyse syntaxique incorrecte.

Une fois regroupés, les résultats extraits restants concernent 245 instances distinctes de sociétés. 46 instances ne sont pas pertinentes[139]. Nous avons comparé les 199 autres résultats obtenus avec ceux du site d'informations financières AlacraStore, qui énumère 195 sociétés (132 rachats et 63 prises de participation). L'application trouve 182 de ces 195 sociétés, plus 17 autres (Finjan Software, Green Button, Changhong…) ayant effectivement fait l'objet d'une opération par Microsoft, mais n'apparaissant pas dans la base de données AlacraStore. Cette méthode offre donc une précision encourageante.

En revanche, le rappel semble largement améliorable, car plusieurs dizaines de phrases du corpus collecté contiennent aussi des occurrences pertinentes qui ne sont pas détectées. L'expérience montre que le rappel de la méthode est directement proportionnel au nombre des paraphrases définies par l'utilisateur du composant ; or, leur création manuelle est un processus chronophage et leur multiplication augmente le temps de calcul à l'exécution.

d) Discussion

Nous estimons que le mécanisme présenté ici a deux forces et deux faiblesses.

D'une part, il tolère la présence de mots intercalés entre ceux qu'on cherche, y compris quand il s'agit de sous-phrases longues (appositions, relatives…), ce qui favorise la précision des résultats. D'autre part, grâce à la couche d'abstraction qu'apporte la plate-forme Antelope, l'ensemble du processus est largement indépendant de la langue et de l'analyseur syntaxique considérés.

En termes de temps de calcul, l'étape limitante reste l'analyse syntaxique, qui représente une opération longue[140] ; en cas d'analyse de corpus important ou de présence d'un nombre élevé de paraphrases, il est impératif d'être sélectif et de définir des filtres en amont pour ne tester que les couples (phrase, patron) qui sont susceptibles de contenir un résultat. D'autre part, comme nous l'avons déjà indiqué, le rappel est améliorable.

e) Perspectives d'amélioration

Un couplage avec le mécanisme de résolution d'anaphores présenté en V.E (page 122) est prévu ; il devrait améliorer les résultats du composant d'extraction d'information en lui permettant de dépasser les limites imposées par le traitement d'une seule phrase. Il devrait également autoriser une diminution du nombre de paraphrases : l'expression « ***Microsoft***$_{\#1}$ *announced* ***it***$_{\#1}$ *bought X* » devient alors inutile, car « *Microsoft bought X* » suffit.

Actuellement, l'étape d'écriture des paraphrases est manuelle. Nous souhaitons ajouter une fonctionnalité d'acquisition semi-automatique de paraphrases[141] en demandant à l'utilisateur un

[139] 16 rumeurs de rachat non avérées (Yahoo!, Disney…), 15 plaisanteries de 1er avril (rachat d'IBM, de l'église catholique…), 12 noms de logiciels appartenant aux sociétés rachetées, et 3 des dirigeants de ces sociétés.

[140] Mais néanmoins de plus en plus rapide, du fait de l'amélioration des algorithmes et de leur implémentation ; nous avons constaté une division par 10 des temps d'analyse syntaxique entre 2006 et 2011, en passant typiquement de quelques secondes par phrase à quelques centaines de millisecondes.

[141] Le lecteur intéressé par cette problématique pourra aussi consulter (Duclaye, 2003).

exemple « bien connu » d'instance d'un prédicat[142] sous la forme d'un n-uplet d'entités nommées. A partir de cette information, il est possible de piloter une recherche de documents sur le Web public des phrases contenant toutes les entités nommées ; ensuite, l'application aux résultats de recherche du mécanisme d'apprentissage de paraphrases présenté en section IV.C.3 (page 68) permettra d'amorcer une liste de variantes, et de proposer à l'utilisateur de valider celles qui lui semblent pertinentes.

f) Conclusion

Nous avons présenté un composant d'extraction d'information robuste, qui met en œuvre une analyse syntaxique d'une façon largement indépendante de la langue. L'intérêt de ce composant vient d'une part de la précision de ses résultats, et d'autre part de la simplicité avec laquelle un utilisateur peut associer différentes paraphrases à un prédicat, grâce à une approche basée sur des exemples.

D. Analyse de sentiments et d'opinions

1. Introduction

L'analyse automatisée d'opinions est une tâche récente qui suscite un intérêt grandissant ; nous montrons en figure 37 la progression[143] entre 2003 et 2011 du pourcentage d'articles d'ACL mentionnant le terme « *sentiment analysis* ». En effet, elle est associée à d'importants enjeux sociétaux et économiques. La promesse est de permettre de comprendre la polarité (positive, neutre ou négative) des avis exprimés sur tel ou tel sujet par (en fonction du contexte) les consommateurs, les citoyens ou les usagers… A l'échelle individuelle, l'analyse de sentiments permet par exemple à une entreprise de déterminer qu'un courrier envoyé par un consommateur mécontent nécessite un traitement prioritaire.

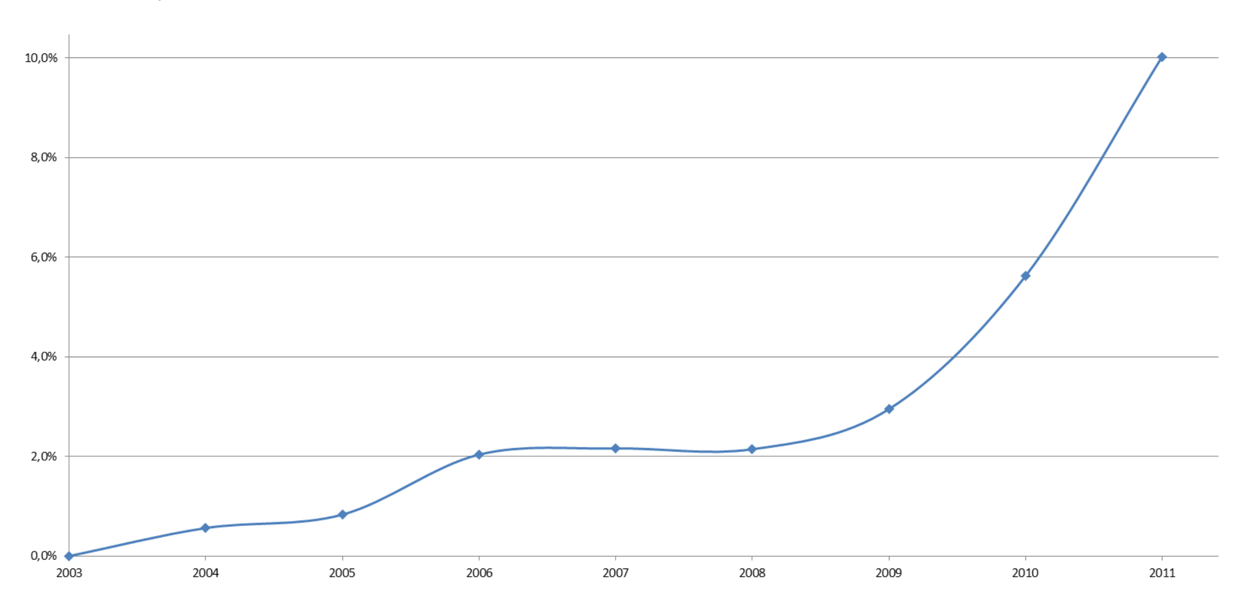


Figure 37 : Progression entre 2003 et 2011 des articles d'ACL mentionnant « *sentiment analysis* »

---

[142] Par exemple, pour le prédicat `acquisition`, en spécifiant acheteur=« *Microsoft* » et societeAchetee=« *Powerset* ».

[143] Pour établir ce graphique, nous avons utilisé l'ACL Anthology Searchbench (http://aclasb.dfki.de), en calculant, pour chaque année, le ratio entre le nombre total d'articles et ceux citant « *sentiment analysis* ».

L'agrégation des avis permet de déterminer les tendances du moment. Les grandes marques comme les organisations politiques rêvent de disposer de sondages à large échelle et en « temps réel » pour prendre le pouls de l'opinion publique. Elles sont très attentives à la maîtrise de leur communication et leur crainte est de souffrir d'une mauvaise réputation en ligne[144].

Ce type d'analyse constitue le champ de recherche traditionnel des sondeurs. Son automatisation est relativement récente. Plusieurs campagnes d'évaluation en fouille d'opinion sont apparues ces dernières années (DEFT, FODOP, SemEval, NTCIR…) et ont tracé les contours de ce qui est techniquement envisageable. Elles ont aussi rappelé la redoutable complexité des problèmes scientifiques soulevés. Les représentations du sens de type « sac de mots » (Cf. section II.A.3) montrent ici leurs limites ; elles peuvent produire le même vecteur termes-fréquences à partir d'énoncés proches, mais sur lesquels l'opinion exprimée diffère sensiblement :

- Ce produit ne vaut rien / Rien ne vaut ce produit.
- Ce produit est de bonne qualité / Ce produit était de bonne qualité.
- Je ne suis pas satisfait de tout / Je ne suis pas satisfait du tout.

La difficulté de l'analyse automatique de sentiments porte aussi bien sur la construction de ressources dédiées (lexique spécifique) que sur l'énumération des situations décrivant un même phénomène (paraphrases, stéréotypes…), voire la définition même de la tâche[145]. S'agit-il d'attribuer une polarité globale à un document entier ? Que faire alors quand des appréciations positives et négatives coexistent dans le même avis ? Comment mesurer l'évolution d'une opinion avec le temps ? Comment restituer une information synthétique pour appréhender d'un coup d'œil des milliers d'avis ? En fonction de la tâche précise à réaliser, il faut distinguer l'objet évalué, l'avis porté dessus et son intensité, déterminer l'émetteur dans le cas d'une conversation multi-locuteurs ou encore le niveau de confiance portée par le locuteur sur son propre avis. Ces facteurs font de l'analyse des sentiments une tâche complexe, avec un décalage important entre les possibilités actuellement offertes par le TAL et la qualité d'une étude humaine.

Reconnaître des entités nommées ou des relations entre ces entités dans du texte tout-venant est déjà complexe du fait des ambiguïtés du langage humain ; mais il ne s'agit ici que d'identifier des éléments factuels. Les phénomènes linguistiques liés à l'expression des sentiments sont nombreux et d'une grande richesse, ce qui en rend l'analyse encore plus complexe. On entre dans un champ subjectif où deux humains peuvent avoir des lectures très différentes d'un même évènement.

Comprendre une opinion exprimée avec de l'humour ou de l'ironie semble aujourd'hui un défi pour la machine. Néanmoins, la recherche progresse rapidement dans ce domaine aussi. (Tsur *et al.*, 2010) propose ainsi l'algorithme SASI (Semi-supervised Algorithm for Sarcasm Identification) destiné à la reconnaissance des sarcasmes dans les avis de consommateurs. Cet algorithme comporte deux

---

[144] Si la notion de veille économique est ancienne, le terme *e-réputation* n'est apparu que récemment, en même temps que le métier de *community manager*.

[145] Dans la préface de la revue TAL (2010, 51.3) dédié aux *Opinions, sentiments et jugements d'évaluation*, (Jackiewicz *et al.*, 2010) propose d'envisager les axes de recherche suivants : *(i) la modélisation linguistique et informatique ainsi que la gestion des données d'opinion (qu'est-ce qu'une « opinion », comment la représenter informatiquement ?) ; (ii) l'expression en langue et en discours (comment les opinions, sous leurs différentes facettes, sont-elles formulées ?) ; (iii) la construction, l'acquisition et la validation des ressources linguistiques ; (iv) les méthodes pour identifier, annoter et extraire automatiquement opinions et sentiments dans des documents textuels ou audiovisuels ; (v) la présentation synthétique de la diversité des opinions.*

étapes : une acquisition semi-supervisée des patrons correspondants et la classification des sarcasmes. L'expérience a été menée sur 66 000 avis sur Amazon (portant sur des livres ou d'autres produits). Les auteurs revendiquent une précision de 77 % et un rappel de 83,1 % pour identifier les phrases sarcastiques.

2. Notre participation à SemEval-2007

a) Objectifs

Le but de la tâche 14 de la campagne SemEval-2007 (*workshop* ACL) était de trouver les sentiments et émotions ressentis par un humain lisant des titres d'articles de presse écrits en anglais. Plus précisément, il fallait reconnaître les six émotions de base (colère, dégoût, peur, joie, tristesse et surprise), et aussi déterminer (point tout aussi complexe[146]) s'il s'agissait globalement d'une bonne ou d'une mauvaise nouvelle.

Antelope nous a permis de construire en une semaine (soit une quarantaine d'heures de développement) le système décrit dans (Chaumartin, 2007a), qui a obtenu à SemEval-2007 la meilleure exactitude (89,43 %) dans la détection des émotions (mais avec un rappel modeste). Une difficulté spécifique dans cette tâche était liée au faible nombre de mots contenus dans chaque titre.

b) Architecture globale

Notre système est principalement basé sur des règles et emploie une approche linguistique. D'un point de vue macroscopique, notre hypothèse est que tous les mots portent potentiellement des émotions dans un titre d'article. Si les ressources linguistiques permettent de détecter ces émotions individuellement, une question qui se pose est comment traiter les titres qui contiennent simultanément des termes positifs et négatifs ; notre approche est d'identifier la tête syntaxique du titre en considérant qu'elle a une importance déterminante.

Nous cherchons également à établir des règles pour détecter des émotions spécifiques. Par exemple, la surprise vient parfois du contraste entre une bonne et une mauvaise nouvelle. Un simple élément lexical est quelquefois caractéristique d'une émotion ; par exemple, une négation ou un modal peut marquer une surprise.

c) Composants utilisés

Notre intuition initiale était qu'une analyse syntaxique du titre faciliterait l'analyse de sentiments. Nos expériences ont montré que nous devions prétraiter le titre pour en faciliter l'analyse syntaxique (Cf. le prétraitement de « décapitalisation » décrit en section e). Comme la plate-forme permet de refaire une même expérience en changeant très facilement d'analyseur (une seule ligne de code est modifiée), nous avons comparé les résultats produits par différents analyseurs pour l'anglais.

Dans les difficultés rencontrées, un titre d'article est parfois réduit à un simple groupe nominal sans verbe. Sur ce type de document, un analyseur basé sur des règles tel que Link Grammar Parser donne clairement des résultats moins bons qu'un analyseur probabiliste comme le Stanford Parser. Ce dernier se révèle plus tolérant avec les constructions grammaticalement imparfaites ; c'est pourquoi nous l'avons choisi pour cette tâche.

---

[146] Par exemple, le titre « *photographe pris en otage au Nigéria et menacé de mort enfin libéré* » contient plusieurs termes négatifs (« *otage* », « *menace* », « *mort* ») et un seul qui est positif dans le contexte (« *libéré* ») ; l'ensemble représente toutefois une bonne nouvelle.

d) Ressources utilisées

Nous avons également utilisé le lexique sémantique, plus précisément WordNet et les ressources dédiées à l'analyse de sentiments : WordNet-Affect et SentiWordNet. Une présentation détaillée de ces deux ressources figure en section IV.B.5, page 61. Nous allons décrire ici comment nous les avons aussi enrichies à cette occasion.

Rappelons que WordNet-Affect (Strapparava, Valitutti, 2004) est une hiérarchie de labels dans le domaine affectif ; les synsets représentant des concepts affectifs sont annotés avec ces labels. Nous avons employé la liste d'émotions du sous-ensemble de WordNet-Affect fourni par les organisateurs de SemEval. Pour l'améliorer, nous avons ajouté manuellement une liste de nouveaux mots (dénotant des émotions) qui nous semblaient pertinents au vu du corpus de test. Par exemple, nous avons associé à l'émotion « peur » des noms (CANCER, DANGER, POVERTY 'pauvreté'…), verbes (DEMOLISH 'démolir', INJURE 'blesser', KIDNAP 'enlever'…), adjectifs (COMATOSE 'comateux', NUCLEAR 'nucléaire', VIOLENT…) et adverbes (DEADLY 'mortellement', WORSE 'pire'…). Le nombre de synsets explicitement associés à chaque sentiment est indiqué dans le tableau 12 : pour chaque partie du discours, la colonne de gauche en donne le nombre initial figurant en standard dans la ressource ; celle de droite représente le nombre de vocables ajoutés par nos soins. En partant des synsets explicitement associés à chaque émotion, notre système a propagé récursivement cette relation aux synsets voisins (en suivant les relations d'hyponymie, de dérivation morphologique, de similarité entre adjectifs et de participe passé).

Sentiment	Noms		Verbes		Adjectifs		Adverbes	
Colère	48	+37	19	+26	39	+16	21	+0
Dégoût	3	+35	6	+19	6	+9	4	+0
Peur	23	+71	15	+26	29	+20	15	+4
Joie	73	+50	40	+22	84	+14	30	+1
Tristesse	32	+88	10	+37	55	+29	26	+4
Surprise	5	+16	7	+29	12	+13	4	+2

Tableau 12 : Nombre de nouveaux vocables ajoutés, par émotion et par partie du discours

Enfin, nous avons utilisé SentiWordNet (Esuli, Sebastiani, 2006) qui, rappelons-le, assigne à chaque synset de WordNet trois valeurs relatives à sa positivité, sa négativité, ou au contraire son objectivité (l'absence de connotation affective)[147]. Ici aussi, nous avons propagé récursivement les scores de positivité et de négativité dans tous les synsets voisins en suivant les relations d'hyponymie (pour les noms et les verbes), de dérivation morphologique et d'antonymie (en échangeant dans ce dernier cas les scores de positivité et de négativité).

e) « Décapitalisation » des mots du titre

Un problème préliminaire que nous avons dû résoudre est lié à l'habitude anglo-saxonne de mettre en majuscule les initiales de tous les mots d'un titre. La première passe de notre système vise ainsi à détecter dans un titre les mots incorrectement capitalisés et à repasser leur initiale en minuscule. Pour cela, nous avons effectué un étiquetage morphosyntaxique du titre avec le SS Tagger (Tsuruoka, Tsujii, 2005). En fonction de la partie du discours à laquelle appartient chaque mot, d'informations

[147] La somme des trois valeurs vaut toujours 1.0 ; par exemple, pour l'adjectif ESTIMABLE#1 on a positivité = 0,75 ; négativité = 0 ; objectivité = 0,25.

trouvées dans WordNet et de quelques heuristiques[148], le système choisit de garder ou non l'initiale inchangée.

L'impact de cette étape préliminaire de transformation est loin d'être négligeable, du point de vue du Stanford Parser. On peut voir, par exemple, le contraste entre l'analyse d'un titre avant (figure 38) et après (figure 39) ce traitement. On remarquera que sur la figure 38, presque tous les mots sont considérés comme étant des noms propres et les dépendances sont incorrectes ; en revanche, sur la figure 39, les mots sont étiquetés avec la bonne partie du discours, et les dépendances sont maintenant presque correctes[149].

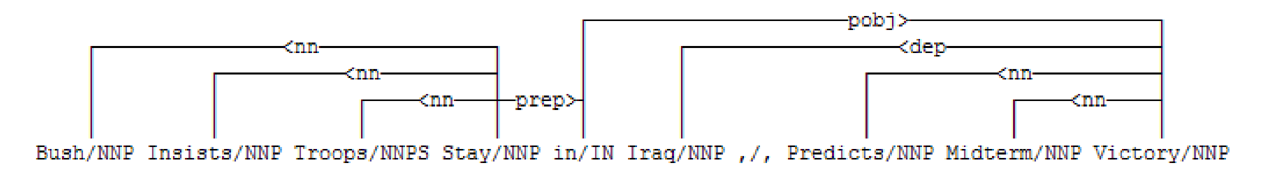


Figure 38 : Sortie du Stanford Parser avec un titre incorrectement « capitalisé »

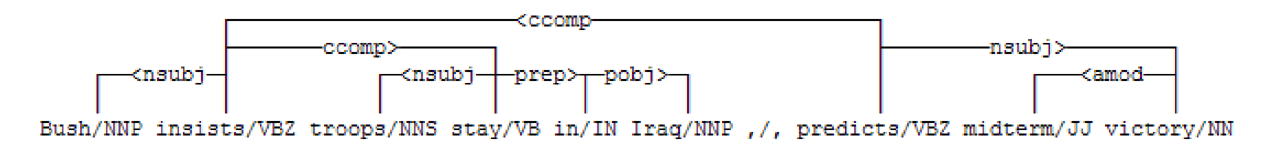


Figure 39 : Sortie du Stanford Parser avec un titre correctement « décapitalisé »

f) Évaluation des mots pris individuellement

À ce stade, nous considérons la sortie du Stanford Parser comme une liste de mots dont la partie du discours est connue. L'utilisation de routines morphologiques permet alors de trouver la forme de base de chaque mot.

Nous n'avons pas essayé de procéder à une désambiguïsation lexicale. Nous avons en effet estimé qu'avec des phrases aussi courtes, peu d'heuristiques pouvaient s'appliquer. Nous avons choisi une autre solution en considérant que la tonalité affective et la valence d'un mot étaient la combinaison linéaire de celles de tous ses sens possibles (pondéré par la fréquence de chaque lemme). Nous avons ainsi calculé la tonalité affective et la valence de chaque mot en utilisant notre version enrichie de WordNet-Affect et SentiWordNet. Nous avons également cherché à détecter certaines informations supplémentaires :

- Une septième émotion, que nous qualifions de « compassion pour des personnes ayant besoin de protection ». Notre hypothèse est que certains mots expriment un besoin implicite de protection. Par exemple, il y a « élève » derrière « école », et « enfant » derrière « adoption ». Ainsi, nous avons établi une liste de mots indiquant une population ayant un besoin naturel de protection ; nous incluons également dans cette liste des mots tels que « troupes », « touristes »…

[148] Par exemple, un mot inconnu de WordNet est probablement un nom propre, auquel cas nous gardons son initiale inchangée.

[149] On remarquera toutefois que l'étiquette `nsubj` de la dépendance entre `predicts` et `victory` devrait être `dobj` ; de même, la dépendance entre `insists` et `predicts` devrait être `coord`.

- Les acronymes dénotant un jargon technologique. Pour ceci, nous avons défini une liste de sociétés technologiques et une expression régulière très simple indiquant qu'un mot (absent de WordNet) contenant des nombres ou des majuscules (hormis l'initiale) représente un élément « high-tech » : cette simple règle semble bien fonctionner sur PS3, iPod, NASA… Nous employons ces indices de contexte high-tech pour augmenter la mesure de la « joie ».
- Des éléments lexicaux que nous pensons être des indicateurs pertinents de la « surprise » : négations, auxiliaires modaux, points d'interrogation.

À ce stade, nous effectuons un post-traitement sur les mots pris individuellement. Quels facteurs causent la colère plutôt que la tristesse ? Notre hypothèse est qu'une volonté humaine maléfique suscite de la colère, alors que des causes naturelles (la maladie, les catastrophes climatiques…) engendrent plutôt la tristesse. Nous avons codé quelques règles utilisant la hiérarchie de noms de WordNet, basée sur l'idée qu'un nom hyponyme d'un synset donné amplifie certaines émotions. Le tableau 13 détaille ces règles.

Le nom hérite-t-il de ?	Émotions à amplifier
UNHEALTHINESS, ATMOSPHERIC PHENOMENON	peur, tristesse
AGGRESSION, HOSTILITY, WRONGFUL CONDUCT	colère, peur, tristesse, dégoût
WEAPONRY, WEAPON SYSTEM	colère, peur, tristesse
UNFORTUNATE PERSON	tristesse, « compassion »
HUMAN WILL	colère

Tableau 13 : Concepts déclenchant l'amplification d'une émotion.

Les émotions détectées servent alors à mettre à jour la valence en augmentant la positivité ou la négativité, comme indiqué dans le tableau 14.

Émotion	Positivité	Négativité
Joie	Augmentation	Diminution
Colère, dégoût, tristesse, peur, « compassion »	Diminution	Augmentation

Tableau 14 : Impact des émotions sur la valence.

g) *Evaluation globale de la phrase*

À ce stade, notre système essaie d'identifier le thème principal du titre. Nous exploitons pour cela l'arbre de dépendances produit par l'analyseur syntaxique. Nous considérons que le mot principal du titre est sa tête syntaxique, c'est-à-dire le mot qui ne dépend d'aucun autre.

Nous pensons que la contribution de ce mot principal est plus importante que celle des autres mots du titre. Dans certains cas néanmoins, nous considérons que la tête lexicale du titre n'en est pas le mot principal ; par exemple, dans un titre commençant par « *une étude dit que* », « *des scientifiques affirment que* », « *la police prétend que* », le mot principal serait la tête de la complétive. Nous multiplions[150] ainsi la valence du mot principal et ses scores d'émotion individuels par 6.

La dernière partie importante du traitement linguistique est la détection des contrastes et des accentuations entre « bonnes » et « mauvaises » choses. Nous recherchons des patrons dans la sortie en dépendances (par exemple un nom sujet d'un verbe ou un nom complément d'objet direct d'un verbe) avec des verbes qui augmentent ou diminuent une quantité ; nous avons « redécouvert »

[150] Ce facteur a été obtenu d'une façon empirique.

ici la notion de *valence shifter*[151] introduite par (Polanyi, Zaenen, 2006). Ceci nous donne la capacité de détecter de très bonnes nouvelles (« *augmente la puissance de réflexion* ») ou de bonnes nouvelles liées à la détérioration de quelque chose de négatif, dont l'importance diminue (« *réduit le risque* », « *ralentit le déclin* », « *l'ouragan s'affaiblit* »…).

h) Résultats

Les résultats de la tâche 14 de SemEval-2007 ont été mesurés avec une mesure de corrélation de Pearson[152]. Notre système, basé sur des règles, détecte les six émotions dans les titres d'articles de presse avec une exactitude[153] moyenne atteignant 89,43 % ; cependant, le rappel est bas. Le tableau 15 détaille ces résultats.

	Corrélation de Pearson		
	Exactitude	**Précision**	**Rappel**
Colère	93,60	16,67	1,66
Dégoût	95,30	0,00	0,00
Peur	87,90	33,33	2,54
Joie	82,20	54,54	6,66
Tristesse	89,00	48,97	22,02
Surprise	88,60	12,12	1,25

Tableau 15 : Résultats de l'annotation des émotions.

Le tableau 16 montre les résultats de détection de la valence. L'exactitude (55 %) est plus faible que dans l'annotation des émotions. Nous attribuons cette différence au fait qu'il est plus facile de détecter des émotions (provenant des contributions individuelles de chaque mot) plutôt que la valence, qui nécessite une compréhension globale de la phrase.

	Corrélation de Pearson		
	Exactitude	**Précision**	**Rappel**
Valence	55,00	57,54	8,78

Tableau 16 : Résultats de l'annotation de la valence.

3. Bilan

En nous inspirant de l'expérience menée pour SemEval, nous avons réalisé d'autres systèmes d'analyse de sentiments pour les appliquer notamment aux avis de consommateurs. La tâche est complexe et son automatisation est aujourd'hui imparfaite, même en déployant des efforts importants pour ajuster le comportement d'un système à une tâche précise. Néanmoins, ses débouchés sont suffisamment attractifs pour encourager la recherche sur le sujet.

Nous estimons que la perspective la plus prometteuse est du côté de l'apprentissage artificiel. En effet, il existe un nombre croissant de sites Web d'avis de consommateurs, sur lesquels on peut

[151] Mot qui fait basculer la valence exprimée par un autre mot.
[152] (Strapparava, Mihalcea, 2007) donne une description détaillée du protocole d'évaluation.
[153] Rappelons que l'exactitude (*accuracy* en anglais) est le pourcentage des éléments bien classés (des vrais positifs et des vrais négatifs) par rapport à l'ensemble de la population. Notons que dans cette évaluation, l'exactitude a été calculée par rapport à toutes les classes possibles ; elle peut donc être artificiellement élevée dans le cas d'ensembles de données asymétriques (comme le sont certaines émotions, en raison du grand nombre de titres neutres). En revanche, la précision et le rappel excluent les annotations neutres.

donner non seulement son avis (sous forme de description textuelle), mais aussi attribuer une note et indiquer les points forts et les points faibles. On a donc là, en principe, un matériau brut de millions de documents permettant d'envisager un apprentissage, du moment qu'on identifie dans ces avis les caractéristiques linguistiques pertinentes.

E. Résolution d'anaphores et de coréférences

1. Introduction

Nous avons vu pour l'instant des tâches opérant au niveau de la phrase. Analyser l'ensemble d'un document nécessite d'effectuer des traitements complémentaires, par exemple de reconnaître le référent d'un pronom ; on parle de résolution d'anaphore lorsque l'on peut retrouver l'antécédent de ce pronom dans le texte qui précède. Le fait de regrouper toutes les références à un même objet est l'extraction d'une chaîne de coréférences. Dans le cadre de la plate-forme, nous avons mis en œuvre un système de ce type. Il a originellement été conçu (et évalué) pour un projet d'extraction de connaissances encyclopédiques. Nous utilisons simultanément des techniques « pauvres en connaissances » et des outils linguistiques plus évolués (analyse syntaxique en profondeur et lexique sémantique).

L'ensemble offre des performances encourageantes sur des articles encyclopédiques. En effet, ces articles possèdent des caractéristiques linguistiques et discursives (topique unique et clair, absence d'humour, etc.) qui permettent d'obtenir des résultats meilleurs que sur d'autres types de textes, tels que des articles de journaux ou des articles scientifiques.

2. Complexité de la résolution d'anaphores

Une *anaphore* est un mot ou un syntagme qui, dans un énoncé, assure la reprise d'un précédent segment appelé *antécédent*. L'utilisation d'anaphore permet d'éviter les répétitions que provoquerait le fait de parler toujours des mêmes entités de la même façon. Dans les exemples figurant dans la suite, les anaphores sont annotées **en gras** et les antécédents identifiés en souligné, comme dans : « L'Amazone est un fleuve très long, seul le Nil **le** dépasse en longueur ».

La résolution d'anaphores (ainsi que son prolongement, l'identification des chaînes de coréférence) est un problème riche en linguistique (pour la modélisation des phénomènes entrant en jeu) et en TAL (pour l'implémentation de ces modèles et la constitution de ressources électroniques). Une vaste littérature existe autour de ce sujet[154], proposant de classifier ces phénomènes (anaphore pronominale, nominale…) ou d'en proposer des modélisations.

Les ambiguïtés sont aussi complexes à lever que dans la tâche de désambiguïsation lexicale. Selon le cas de figure, on doit faire appel à des connaissances de nature lexicale, syntaxique, sémantique et pragmatique, ainsi qu'à une compréhension du contexte. Par exemple, les trois phrases suivantes[155] partagent la même construction syntaxique ; seul l'adjectif final varie, produisant à chaque fois une interprétation différente du pronom **ils** :

[154] Voir par exemple (Salmon-Alt, 2002) ou (Kleiber, 1994).
[155] Notons que nous ne proposons pas de résoudre des cas aussi complexes, qui font appel à une connaissance du monde évoluée.

- Les gardiens ont donné les fruits aux singes parce qu'**ils** étaient *pourris*.
- Les gardiens ont donné les fruits aux singes parce qu'**ils** étaient *affamés*.
- Les gardiens ont donné les fruits aux singes parce qu'**ils** étaient *rassasiés*.

3. Méthodes de résolution d'anaphores pronominales

Les principales méthodes de résolution d'anaphores pronominales ne sont pas récentes. (Hobbs, 1978) propose un algorithme[156] utilisant l'analyse syntaxique d'un texte. (Lappin, Leass, 1994) proposent un algorithme en plusieurs étapes[157] nécessitant une analyse syntaxique, et revendiquent une précision de 86 % sur un corpus technique en anglais[158]. (Mitkov, 1998) présente une approche « robuste et pauvre en connaissance », basée sur plusieurs heuristiques qui exploitent une analyse syntaxique superficielle ; évaluée sur un corpus de manuels techniques (en anglais, polonais et arabe), cette approche donne une précision de l'ordre de 90 %.

4. Algorithme mis en œuvre

Antelope propose un composant de résolution d'anaphores et d'identification de chaînes de coréférence. Ce composant fonctionne indifféremment en anglais ou en français avec une version unique ; les spécificités de chacune de ces deux langues sont prises en compte par une dizaine de lignes de code seulement.

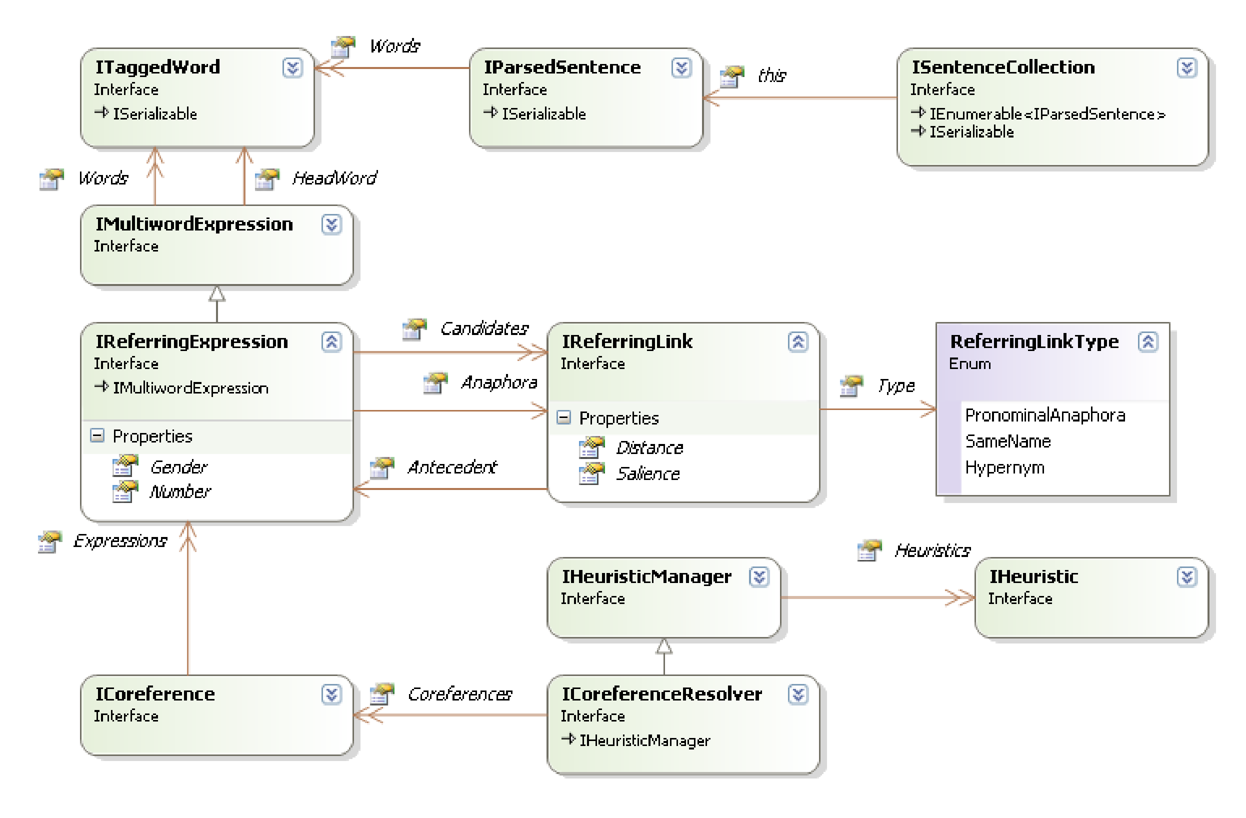


Figure 40 : Modèle de programmation pour la résolution d'anaphores

[156] Etant donné un pronom dans une phrase, l'algorithme effectue un parcours de l'arbre syntaxique de la phrase (et éventuellement de la phrase précédente) à la recherche de son antécédent.
[157] Identification des pronoms pléonastiques ; algorithme de liage identifiant l'antécédent d'un pronom réfléchi ou réciproque dans la même phrase ; assignation d'une valeur de saillance pour chaque syntagme nominal.
[158] Précisons toutefois que l'analyse syntaxique a été corrigée manuellement.

La figure 40 présente les classes mises en œuvre par notre composant ; la classe CoreferenceResolver joue le rôle de chef d'orchestre local (c'est-à-dire sans interaction avec d'autres composants) ; une expression référentielle (ReferringExpression) est un cas particulier d'expression multi-mots qui gère les ambiguïtés avec une liste de candidats possibles dont un seul sera retenu en tant qu'anaphore.

Nous utilisons simultanément des heuristiques classiques, pauvres en connaissances, qui s'appliquent dès le niveau d'étiquetage morphosyntaxique, et des techniques requérant une analyse syntaxique profonde ainsi que le lexique sémantique. A un niveau macroscopique, notre algorithme enchaîne d'une façon classique les opérations suivantes :

- Analyse syntaxique du texte (étiquetage morphosyntaxique ou analyse syntaxique[159]).
- Parcours du document :
    - Détection des pronoms personnels et possessifs.
    - Détermination du caractère anaphorique du pronom, par élimination de chaque *il* pléonastique (« *It is possible that…* »[160]) ou impersonnel (« *il pleut…* »)[161].
- Pour les pronoms anaphoriques :
    - Marquage des différents antécédents candidats.
    - Vérification des contraintes syntaxiques (c-commande).
    - Vérification de l'accord en genre et en nombre.
    - Application de différentes heuristiques qui augmentent ou diminuent le score de chaque candidat ; celui présentant au final le score le plus élevé est retenu.
- Extraction des chaînes de coréférences par calcul des composantes connexes du graphe des anaphores.

Les heuristiques sont le cœur de traitement de ce composant. Notre première implémentation utilisait les heuristiques de (Mitkov, 1998)[162]. Disposant d'une plate-forme autorisant une analyse syntaxique, nous avons pu y ajouter les caractéristiques de l'algorithme de (Lappin, Leass, 1994). Nous avons également implémenté une heuristique de résolution des anaphores nominales, qui permet, par exemple, d'identifier correctement l'anaphore dans « *As **Lincoln** sat in the balcony, Booth crept up behind the **President**'s box.* » Cette dernière heuristique utilise les mesures de similarité décrites page 79.

5. Évaluation et perspectives

Le composant a été évalué dans le cadre d'articles d'encyclopédies. Ces articles ont quelques caractéristiques qui facilitent leur analyse automatique : ils sont (généralement) correctement écrits dans un style concis, sans humour ; ils relatent des faits, avec des temps de verbe le plus souvent au passé. Les anaphores étant fortement présentes dans de tels articles, leur résolution est indispensable si on souhaite parvenir à une représentation sémantique correcte d'un article. Ces anaphores sont (majoritairement) pronominales, et portent (le plus souvent) sur le titre de l'article,

[159] En utilisant la sortie (graphe de dépendances) produite par un analyseur syntaxique.
[160] Approximativement 3% des « *it* » (mesure sur 20 articles choisis au hasard dans la Wikipedia en anglais).
[161] Nous nous sommes inspiré pour cela de (Danlos, 2005) qui montre comment, dans ce cas précis, un traitement peut atteindre une précision remarquablement élevée (97,5%).
[162] Obliqueness, definiteness, lexical reiterations, section heading, referential distance, boost pronouns, collocation match, parenthesis…

c'est-à-dire son sujet[163]. La figure 41 montre l'identification des chaînes de coréférences calculée par notre composant sur l'article « Nile River » de l'encyclopédie en ligne Britannica (version 2004) ; chacune d'entre elles est affichée avec une couleur distincte[164].

The longest **river** in the world, **it** is about 4132 miles (6650 km) long from **its** remotest headstream and 3473 miles (5588 km) from Lake_Victoria to the **Mediterranean_Sea**. **It** flows generally north from eastern Africa through Uganda, The Sudan, and **Egypt**. **It** receives major tributaries, including the Blue_Nile and the **Atbara_River**, before entering **Lake_Nasser** near the Egypt-Sudan border. After the **Aswan_High_Dam** impounds the **lake**, **it** continues northward to **its** delta near Cairo, where **it** empties into the **Mediterranean**. The first use of the **Nile** for irrigation in **Egypt** began when seeds were sown in the mud left after **its** annual floodwaters had subsided. **It** has supported continuous human settlement for at_least 5000 years, with canals and waterworks built in the 19th century. The **Aswan_High_Dam**, built in 1959-- 70, provides flood protection, hydroelectric power, and a dependable water_supply for both crops and humans. The **Nile** is also a vital waterway for the transport of people and goods.

Figure 41 : Identification des chaînes de coréférences sur un article portant sur le Nil

Sur quarante-sept anaphores relevées lors d'une annotation manuelle de onze articles, quarante-six ont été détectées correctement. L'antécédent a été correctement trouvé dans quarante-trois cas. Sur ce jeu de tests réduit, la précision est donc de 93 % et le rappel de 97 %.

F. Regroupement de documents

1. Introduction

Le regroupement de documents (*clustering* en anglais) permet de partitionner un corpus en sous-ensembles présentant des similitudes. Une telle opération porte donc non plus sur un document individuel mais sur la globalité d'un corpus. Elle offre de l'intérêt pour plusieurs applications de TAL. Nous souhaitions donc intégrer en standard un composant de regroupement dans la plate-forme. Nous présentons ici un état de l'art (non exhaustif mais représentatif des différentes voies possibles) et des précisions sur notre implémentation de deux algorithmes : Bron-Kerbosch et Spectral Clustering.

2. Préambule à propos des implémentations

L'information fournie en entrée à un algorithme de regroupement est une matrice termes-documents. C'est une matrice dont les colonnes représentent les documents, et les lignes les termes du corpus ; chacune des cellules compte le nombre de fois où un terme donné apparaît dans un document donné. Cette matrice peut devenir très volumineuse : par exemple, un corpus de 200 000 documents, écrits avec 30 000 termes distincts, représenterait une matrice de six milliards de cellules ; l'espace mémoire nécessaire pour stocker un tel tableau bidimensionnel (avec des entiers 32 bits) serait donc de 24 giga-octets, ce qui excède la capacité mémoire usuelle des ordinateurs actuels. Heureusement, tous les termes du corpus ne se retrouvent pas dans chaque document ;

[163] Nous pourrions nous risquer à proposer comme algorithme non-subtil de résolution d'anaphores dans le texte de l'article, un brutal *recherche & remplace* du pronom qui y apparaît le plus fréquemment, par son titre. Un tel algorithme pourrait servir de baseline dans ce cas particulier.

[164] On y remarquera une erreur : *Atbara River* se retrouve coréférent avec *Nile River*.

pour des corpus larges, seuls 1 % à 5 % des termes vont effectivement se retrouver dans un document donné ; cette matrice est donc surtout remplie de zéros. Le stockage d'une matrice termes-documents nécessite donc une structure de données adaptée, appelée « matrice creuse ». L'idée est de n'y stocker que les entrées non nulles de la matrice pour économiser la mémoire utilisée par rapport à une structure naïve de tableau[165].

Les manipulations de telles matrices doivent aussi être implémentées soigneusement. Par exemple, l'algorithme naïf de multiplication matricielle produirait une matrice pleine en traitant deux matrices creuses. Nous avons initialement utilisé la librairie Colt, développé par le CERN en Java, qui permet de faire des calculs sur des matrices creuses de façon optimisée ; l'équipe Proxem a par la suite implémenté une librairie de calcul améliorant ces optimisations.

3. Etat de l'art

Le regroupement consiste à partitionner un ensemble de documents sans connaître à l'avance le nombre de partitions ni leur nature[166]. L'objectif du problème de regroupement est d'obtenir des groupes compacts et homogènes, et que ces groupes soient aussi différents que possible entre eux ; en termes plus formels, cela revient à minimiser l'inertie intra-classe[167] et maximiser l'inertie inter-classes[168]. Plusieurs techniques ont été proposées pour arriver à partitionner les données en se basant le plus souvent sur des espaces vectoriels euclidiens, où un document est représenté par un vecteur de termes. Les algorithmes de regroupement de documents que nous verrons dans la suite se basent sur ce modèle.

a) Regroupement hiérarchique

Le regroupement hiérarchique ascendant (Hastie *et al.*, 2001) consiste à considérer dans un premier temps que chaque document forme sa propre classe. Ensuite, on regroupe deux par deux les classes qui sont les plus proches, jusqu'à obtenir une classe unique (Cf. figure 42). Pour exprimer la distance entre les classes, des critères comme celui de Ward (qui maximise l'inertie inter-classes) ont été proposés. Ensuite l'arbre résultant peut être coupé selon un certain seuil d'inertie intra-classe.

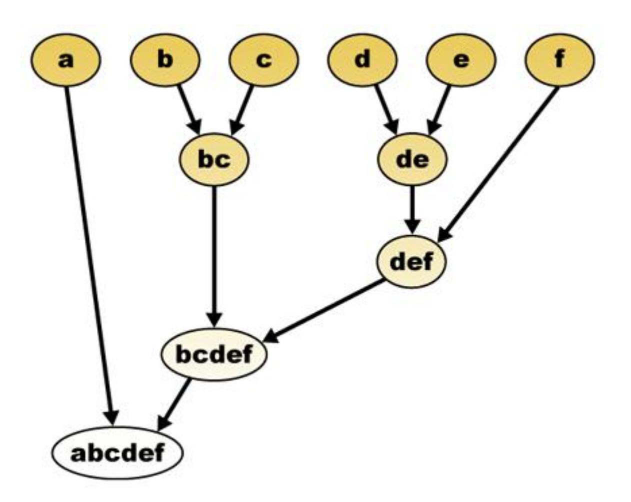


Figure 42 : Un exemple de regroupement hiérarchique

---

[165] Un exemple de représentation de matrice creuse est le format *Yale Sparse Matrix*.
[166] On parle de classification ou de discrimination quand ces dernières informations sont connues *a priori*.
[167] L'inertie intra-classe est la variance des points d'un même groupe.
[168] L'inertie inter-classes est la variance des centres des groupes.

Le regroupement hiérarchique peut également être descendant : tous les éléments sont initialement considérés appartenir à une même classe ; on cherche ensuite à séparer cette classe en deux en maximisant l'inertie inter-classes. De même, l'arbre résultant peut être coupé selon un certain seuil d'inertie intra-classe.

b) Regroupement par cliques (ou regroupement flou)

Le corpus est ici considéré comme un graphe ; les documents sont interconnectés, les arcs du graphe représentant la similarité entre ces documents. Le regroupement par cliques cherche à regrouper les éléments qui se trouvent dans une même clique, du point de vue de ce graphe. L'avantage est que le regroupement n'est pas strict, un document pouvant appartenir à plusieurs classes ; c'est pourquoi on le qualifie aussi de regroupement flou.

c) Regroupement QT

Le regroupement QT (*Quality Threshold*) (Heyer *et al.*, 1999) consiste à chercher pour chaque élément un regroupement possible qui ne dépasse pas un diamètre fourni par l'utilisateur, puis à choisir le regroupement contenant le plus d'éléments. Cette phase est répétée sur les éléments qui ne sont pas dans ce regroupement jusqu'à ce que le plus grand regroupement soit inférieur à un seuil fourni par l'utilisateur. Cette technique permet de trouver des grandes classes en ayant une qualité intra-classe satisfaisante.

d) Regroupement spectral

Comme dans le regroupement par cliques, le problème peut être reformulé en termes de graphe. On considère un graphe pondéré non orienté où les sommets correspondent aux documents, et les arêtes sont pondérées selon la ressemblance entre deux documents. Le problème est alors de trouver une partition du graphe telle que les classes soient aussi différentes que possible, avec des documents similaires entre eux au sein d'une même classe.

La théorie spectrale des graphes (Chung, 1997) montre que la recherche d'une partition d'un graphe en k classes revient à la recherche des k plus grands vecteurs propres (au sens de leurs valeurs propres) de la matrice laplacienne du graphe. Pour éviter d'avoir à fournir explicitement une valeur arbitraire de k, plusieurs techniques ont été proposées pour trouver un partitionnement satisfaisant du graphe :

- Une première technique naïve est de calculer pour toutes les valeurs de k le partitionnement qui minimise la valeur de la coupe du graphe. Le problème de cette méthode est le temps de calcul sur des corpus importants.
- Une seconde approche, inspirée du regroupement hiérarchique descendant, est de bipartitionner le graphe jusqu'à ce que la coupe soit en dessous d'un certain seuil. Cette idée est utilisée dans l'algorithme NCut décrit dans (Shi, Malik, 1997) et amélioré dans (Ding *et al.*, 2001). Le problème de cet algorithme est que le partitionnement s'arrête au niveau du seuil.
- Une troisième technique a vu le jour pour contourner ce genre de problème. Elle consiste à construire entièrement la hiérarchie du graphe en le bipartitionnant puis à regrouper les nœuds dans l'arbre résultant avec un critère d'agrégation. L'algorithme décrit dans (Cheng *et al.*, 2006) utilise ce principe.

e) *Réduction de dimensions*

Le problème le plus important dans le regroupement de documents provient de la dimension élevée des matrices termes-documents. Pour pallier ce problème, des techniques visant à réduire la dimension de l'espace ont été introduites. La recherche des composantes principales vise à ne retenir que les axes qui contiennent le plus d'informations. Le regroupement est alors amélioré car une partie du bruit (l'information non pertinente) a été supprimée. Cependant, le fait que les composantes principales (qui correspondent *a priori* aux différents sujets) soient orthogonales pose un problème dans les regroupements de documents, car des sujets différents ne sont pas forcément indépendants (par exemple, la biologie et l'informatique ont en commun la bio-informatique).

La réduction de dimension peut également être utilisée en tant que telle pour classifier des documents en utilisant la factorisation matricielle non négative (*Nonnegative Matrix Factorization*) présentée dans (Xu *et al.*, 2003). Cette factorisation consiste à calculer une approximation de la matrice termes-documents comme le produit de deux matrices (l'une en fonction des termes, l'autre en fonction des documents) représentant les différentes classes. L'avantage de cette réduction de dimensions est que les classes produites ne sont pas forcément orthogonales comme dans la recherche des composantes principales.

4. Implémentations dans la plate-forme

Nous avons implémenté deux algorithmes dans la plate-forme, un pour le regroupement par cliques (algorithme de Bron-Kerbosch) et un pour le regroupement spectral. Les deux offrent de bonnes performances.

a) *Regroupement par cliques (Bron-Kerbosch)*

Notre implémentation de Bron-Kerbosch est inspirée par (Cazals, Karande, 2008), dont l'algorithme est décrit en figure 43. Notre implémentation utilise aussi une fonction qui transforme une matrice de similarité en une matrice d'adjacence en utilisant une valeur de seuil pour supprimer les éléments qui sont trop dissemblables.

```
Algorithm call: IK_*(∅, V[G], ∅).

IK_*(R, P, X)
if P = ∅ and X = ∅ then
    Report R as a maximal clique
else
    Let u_p be the pivot vertex //see text
    Assume P = {u_1, u_2, ..., u_k}
    for i ← 1 to k do
        if u_i is not a neighbor of u_p then
            P = P − {u_i}
            R_new = R ∪ {u_i}
            P_new = P ∩ N[u_i]
            X_new = X ∩ N[u_i]
            IK_*(R_new, P_new, X_new)
            X = X ∪ {u_i}
```

Figure 43 : Algorithme de Bron-Kerbosch

L'avantage du regroupement par cliques est de permettre d'avoir simplement un regroupement flou, où un élément peut appartenir à plusieurs classes. Ceci est intéressant pour le lexique sémantique, puisqu'un sens donné peut être regroupé dans plusieurs sens macroscopiques. Dans ce contexte, notre implémentation est « raisonnablement » rapide (quelques millisecondes pour trouver les cliques dans un graphe d'une centaine d'éléments).

b) Regroupement spectral

L'avantage du regroupement spectral est de dispenser l'utilisateur de devoir fournir un seuil *a priori*. En effet, ces seuils sont parfois difficiles à trouver pour avoir un bon regroupement.

L'algorithme de regroupement spectral procède en deux phases. La première phase (division) consiste à créer un regroupement hiérarchique en bipartitionnant récursivement le graphe résultant de la matrice de similarité. La seconde phase (fusion) cherche le meilleur regroupement arborescent suite à la phase de division. La figure 44 illustre le principe de l'algorithme. Son fonctionnement précis est détaillé en annexe, page 202 ; le point très intéressant de l'algorithme est qu'il tient compte du fait que la matrice est creuse pour optimiser les calculs.

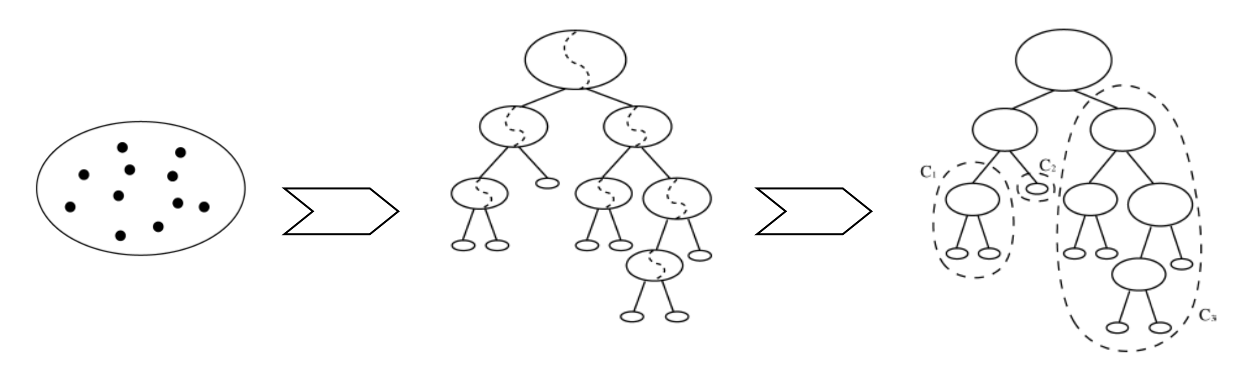


Figure 44 : Exemple simplifié de mise en œuvre de l'algorithme de regroupement spectral

Les performances de notre implémentation initiale étaient (subjectivement) correctes : le regroupement de 400 documents courts (deux lignes de texte) nécessitait 2 secondes de temps de calcul. Depuis, l'équipe Proxem a optimisé cette implémentation en tombant à moins d'une seconde de calcul sur le même type de corpus.

5. Applications

Voyons quelques exemples concrets de mise en œuvre du regroupement dans des applications utilisant Antelope, ou dans la plate-forme elle-même.

Une offre d'emploi est souvent reprise en plusieurs exemplaires, avec des légères variantes par rapport à l'offre initiale. Pour déterminer le nombre de postes ouverts à un instant donné (qui est sensiblement inférieur au nombre d'offres), il faut donc procéder à un dédoublonnement du corpus d'offres.

Autre exemple : nous avons vu dans la partie dédiée au lexique sémantique que le découpage des sens proposé par WordNet est parfois trop fin. Nous avons appliqué des algorithmes de regroupement aux définitions des sens des différents vocables pour les fusionner en sens macroscopiques, de façon à simplifier la désambiguïsation lexicale.

Enfin, dans une application de recherche d'information, la visualisation d'un grand nombre de résultats est souvent problématique. Il est donc pratique de regrouper les résultats en sous-ensembles cohérents. Nous illustrerons ce point dans la partie qui présente les applications d'Antelope, notamment aux sections VI.B.3 (partie 134) et VI.D.5 (page 144).

6. Conclusion

Nous prévoyons d'utiliser dans le futur des algorithmes incrémentaux (ou algorithmes *online*) qui modifient dynamiquement les regroupements déjà créés lors de l'ajout de nouveaux documents. Ils évitent en effet de tout recalculer et permettent ainsi de traiter de grands volumes de données en constante augmentation.

Nous envisageons aussi d'implémenter l'algorithme Lingo (Osiński *et al.*, 2004). Il trouve d'abord les descriptions qui pourraient s'appliquer aux documents en réduisant les dimensions, puis assigne ces documents à chacune des descriptions trouvées.

Partie VI. Applications

Plusieurs applications du TAL ont produit des logiciels largement utilisés. Parmi les plus connues, on peut citer : la traduction automatique (application pionnière du TAL), la correction orthographique ou grammaticale, la recherche d'information (moteurs de recherche), la reconnaissance vocale, la synthèse de la parole, la génération automatique de textes, le résumé automatique de textes. En plus de ces applications historiques, le TAL trouve aujourd'hui sa place en entreprise dans des domaines très divers. La classification de documents peut ainsi servir au routage automatique de documents entrants vers les bons destinataires. D'autres applications mettent en œuvre une extraction d'information sur mesure, pour analyser par exemple des opinions exprimées par des consommateurs ou des décisions de justice.

Nous présentons d'abord trois applications que nous avons développées avec l'équipe Proxem. Elles intègrent tout ou partie des composants d'Antelope, avec une complexité croissante qui reflète une progression chronologique [169]. Les deux premières concernent l'extraction d'information. Le chapitre A présente un sous-ensemble du projet SCRIBO ; le composant de reconnaissance d'entités nommées (Cf. V.B.5, page 99) est appliqué aux articles de presse, en respectant le standard UIMA (Cf. III.H, page 35). L'outil de veille économique décrit au chapitre VI.B (page 134) utilise en plus les composants d'extraction de relations (Cf. V.C, page 106) et de regroupement de documents (Cf. V.F, page 125).

La troisième application s'appelle Ubiq. Développée par l'équipe Proxem, elle exploite tous les composants de la plate-forme Antelope. Ubiq est une solution d'aide à la décision, aujourd'hui déclinée en deux versions. La première, présentée au chapitre D, concerne l'e-réputation par analyse d'avis de consommateurs. La seconde, détaillée au chapitre E, porte sur les documents RH ; après une analyse sémantique, elle permet de trouver les meilleurs CV correspondant à une offre ou les meilleurs postes pour un profil donné. Dans ces deux cas, nous avons dû procéder en amont à l'acquisition de connaissances spécifiques au domaine traité ; c'est pourquoi nous commencerons par présenter au chapitre C notre démarche pour enrichir le lexique sémantique sur un domaine.

Ces applications relèvent plus du projet d'ingénierie ou de la recherche appliquée que de la recherche fondamentale[170]. Nous les présentons pour illustrer concrètement le résultat de la thèse : la capacité à créer rapidement des applications où le TAL joue un rôle central, et qui rend un service tangible à des utilisateurs qui n'ont aucune idée de ce qu'est le TAL[171].

Pour finir, nous tenons à montrer qu'il n'y a pas que l'équipe Proxem qui met en œuvre Antelope. Le chapitre F présente une dizaine de projets de recherche qui ont utilisé la plate-forme.

---

[169] SCRIBO s'est déroulé de 2008 à 2010. Le développement de la version d'Ubiq pour l'analyse des avis de consommateurs a démarré en 2010 ; celui de la version dédiée aux ressources humaines a débuté en 2011.

[170] Néanmoins, certains des composants d'analyse ont été améliorés pour tenir compte du contexte applicatif.

[171] Nous oserons le parallèle suivant : si un beau moteur (Antelope) est la partie technologique la plus noble d'une voiture (Ubiq), sa principale qualité est de savoir se faire oublier au quotidien, au profit du tableau de bord et de la carrosserie (interface homme-machine simple à utiliser, rapports synthétiques compréhensibles).

A. Extraction d'information dans des articles de presse (projet SCRIBO)

1. Objectif

SCRIBO (*Semi-automatic and Collaborative Retrieval of Information Based on Ontologies*) est un projet collaboratif de recherche appliquée en informatique, en linguistique et en ingénierie des connaissances, qui s'est déroulé de mi-2008 à fin 2010. Ce projet a été labellisé par le groupe de travail Logiciel Libre du pôle de compétitivité Systematic. Son objectif était la mise au point d'algorithmes et d'outils collaboratifs libres pour l'extraction de connaissances à partir de textes ou images et l'annotation semi-automatique de documents numériques. Les principaux acteurs du projet SCRIBO sont le CEA LIST, l'INRIA, le LRDE (EPITA), Nuxeo, Proxem, Tagmatica et XWiki, ainsi que l'AFP (Agence France Presse) et Mandriva en tant qu'entreprises utilisatrices pilotes.

L'AFP a mis en œuvre les composants SCRIBO dans le contexte de l'annotation semi-automatique de flux d'informations multimédia multilingues, aussi bien dans des domaines généraux que thématiques, ainsi que dans un contexte de veille. Mandriva a expérimenté les composants SCRIBO sur deux chantiers : d'une part pour procéder à l'annotation automatique de la documentation du système d'exploitation Mandriva Linux (manuels techniques, questions-réponses, articles de presse, interviews, etc.) dans le but d'améliorer l'accès à des informations spécifiques dans différentes langues ; d'autre part pour enrichir les fonctionnalités du bureau sémantique KDE.

2. Reconnaissance d'entités nommées

SCRIBO était subdivisé en plusieurs sous-projets. L'un d'eux, piloté par Proxem, consistait en l'acquisition de connaissances depuis des documents textuels. L'objectif du sous-projet était notamment la détection des personnes, lieux, organisations et montants monétaires cités dans les dépêches de l'AFP.

a) Compatibilité des annotations avec UIMA

Le projet SCRIBO a permis d'étendre la plate-forme Antelope, d'une part en créant une première implémentation de la reconnaissance d'entités nommées, et d'autre part en rendant ses résultats conformes au standard UIMA (Cf. chapitre III.H, page 35). En effet, comme plusieurs éditeurs de composants de TAL participaient au projet, cette architecture a été retenue pour partager les annotations provenant de différents composants. La fusion de ces différents jeux d'annotations (éventuellement en contradiction) relevait de la responsabilité de l'AFP.

b) Résultats

En utilisant la convention IOB (Cf. V.B.5.b), on cherche à attribuer les étiquettes B_Personne, I_Personne, B_Lieu, I_Lieu, B_Organisation, I_Organisation, B_Monnaie, I_Monnaie et O. Les caractéristiques prises en compte sont la forme de base du mot, sa partie du discours ainsi que la présence de majuscules et de nombres. Le corpus de test était constitué de 130 articles.

Nous avons mené deux expériences d'apprentissage en utilisant les CRF, en considérant les caractéristiques du mot courant combinées à celles des *n* mots précédents et suivants au sein d'une fenêtre de taille 2 et 5. Le tableau 17 détaille les résultats mesurés sur la fenêtre de taille *n*=2, et le tableau 18 ceux de la fenêtre de taille *n*=5. Dans ces tableaux, à chaque étiquette sont associées le

nombre de mots annotés dans le corpus de test (colonne #ref), le nombre de mots reconnus par le CRF (colonne #model) et le nombre de mots annotés dans le corpus de test également reconnus par le CRF (colonne #match). On constate que la qualité de reconnaissance varie selon le type d'entité ; la comparaison des deux tableaux montre que l'élargissement de la fenêtre de 2 à 5 n'améliore pas la F-mesure.

Etiquette	#ref	#model	#match	précision	Rappel	F-mesure
O	44137	44350	43887	0.9896	0.9943	0.9919
B_Personne	678	644	538	0.8354	0.7935	0.8139
I_Personne	663	635	551	0.8677	0.8311	0.8490
B_Lieu	1082	1052	993	0.9439	0.9177	0.9306
I_Lieu	185	191	159	0.8325	0.8595	0.8457
B_Organisation	225	142	113	0.7958	0.5022	0.6158
I_Organisation	148	108	71	0.6574	0.4797	0.5547
B_Monnaie	40	38	38	1.0000	0.9500	0.9744
I_Monnaie	81	79	79	1.0000	0.9753	0.9875
Moyenne				**0.8802**	**0.8114**	**0.8404**

Tableau 17 : Résultats de la reconnaissance d'entités nommées avec une fenêtre de taille 2

Etiquette	#ref	#model	#match	précision	rappel	F-mesure
O	44137	44338	43882	0.9897	0.9942	0.9920
B_Personne	678	641	539	0.8409	0.7950	0.8173
I_Personne	663	625	543	0.8688	0.8190	0.8432
B_Lieu	1082	1060	1002	0.9453	0.9261	0.9356
I_Lieu	185	178	157	0.8820	0.8486	0.8650
B_Organisation	225	154	121	0.7857	0.5378	0.6385
I_Organisation	148	130	71	0.5462	0.4797	0.5108
B_Monnaie	40	36	36	1.0000	0.9000	0.9474
I_Monnaie	81	77	77	1.0000	0.9506	0.9747
Moyenne				**0.8731**	**0.8056**	**0.8360**

Tableau 18 : Résultats de la reconnaissance d'entités nommées avec une fenêtre de taille 5

Nous avons effectué une fouille d'erreur sur les nouvelles instances d'entités reconnues par le CRF mais qui n'étaient pas annotées dans le corpus de test, pour déterminer si elles représentaient effectivement des entités nommées. Une vérification manuelle confirme que ce n'est pas le cas. Par exemple, certaines personnes ou organisations sont détectées comme étant des lieux. Ces erreurs sont dues d'une part à une prédiction incorrecte du CRF, mais aussi aux erreurs d'annotation du corpus d'apprentissage. Le tableau 19 résume ces résultats par classe d'entité.

Entité nommée	Entités nouvelles détectées par le CRF	Entités valides
Personne	80	45
Lieu	42	20
Organisation	22	15
Monnaie	1	1

Tableau 19 : Résultats de la fouille d'erreur sur les entités nouvelles proposées par le CRF

B. Veille économique sur le Web

1. Objectif

Un outil d'extraction d'information est livré avec la plate-forme à titre de démonstration. Il utilise les composants de segmentation, d'extraction de relations après analyse syntaxique, de reconnaissance d'entités nommées et de regroupement de documents.

Cet outil permet d'automatiser des requêtes de veille économique sur le Web pour y trouver des événements reliant des entités nommées, comme par exemple le rachat d'une société par une autre, le lancement d'un nouveau produit par une entreprise, la nomination ou le départ d'un dirigeant…

2. Mode opératoire

L'utilisateur de l'outil exprime d'abord ses requêtes sous forme de plusieurs paraphrases de l'événement qu'il recherche. La figure 36 (page 113) illustre l'exemple « rachat de société » avec onze réalisations linguistiques en anglais. Ces paraphrases permettent ensuite d'effectuer des requêtes sur un moteur de recherche du Web[172]. Les adresses obtenues sont « dédoublonnées », puis chaque page HTML est chargée et examinée. L'outil filtre les phrases et retient celles qui contiennent, dans le même ordre, les mots-clés de l'une des paraphrases de la requête. La méthode d'extraction d'information utilisée s'appuie sur une traduction des paraphrases en patrons syntaxiques ; les phrases susceptibles de contenir l'information recherchée font l'objet d'une analyse syntaxique en dépendances, puis d'un appariement de formes avec les graphes syntaxiques des patrons (présenté en section V.C.1.b). Cette méthode fournit donc des résultats précis, mais au prix d'un temps de calcul élevé (une heure sur cet exemple, en incluant la collecte Web).

Notons que cet outil peut s'interfacer avec Protégé (éditeur d'ontologie au format OWL[173]). L'utilisateur y associe directement des paraphrases à une classe de relations, en utilisant le mécanisme d'annotations de Protégé. Le composant d'extraction d'information importe ces annotations à l'aide d'une interface de programmation qui permet de lire l'ontologie au format OWL. Une fois les données extraites, elles peuvent être exportées au format RDF et venir enrichir l'ontologie avec de nouvelles instances.

3. Visualisation des résultats

Sur l'exemple précité utilisant onze paraphrases, la recherche des rachats effectués par Microsoft collecte d'abord 2 160 pages Web. Le composant d'extraction d'information affichait initialement un résultat brut sous la forme d'une liste à plat de 1 353 noms, comme montré en figure 45, où plusieurs lignes peuvent faire référence à une même société.

[172] Actuellement Microsoft Bing ou Yahoo proposent une interface de programmation applicative permettant de piloter des recherches Web à partir d'un programme ; Google en a aussi offert une jusqu'à novembre 2010.
[173] *Web Ontology Language* : langage de modélisation d'ontologies basé sur les logiques de description, et standard du Web sémantique (Cf. page 186).

results	text
buyer=Microsoft, company=mobile advertising fir...	Microsoft acquires mobile advertising firm ScreenTonic
buyer=Microsoft, company=mobile advertising co...	Microsoft buys mobile advertising company ScreenTonic
buyer=Microsoft, company=mobile ad firm Screen...	Microsoft buys mobile ad firm ScreenTonic
buyer=Microsoft, company=MobiComp	Microsoft Acquires MobiComp
buyer=Microsoft, company=Million	we learned that Microsoft bought 1. 6% of Facebook for $240 Million.
buyer=Microsoft, company=Microsoft Business So...	Axapta is one of the four ERP suites offered by Microsoft Business Solutions, a subsidiary of Micro...
buyer=Microsoft, company=Microsoft . '	" Microsoft Acquires ' Microsoft Acquires. "' It is the first " Microsoft Acquires" Collage. Enjoy!
buyer=Microsoft, company=MessageCast	Microsoft Acquires MessageCast
buyer=Microsoft, company=MessageCast	Microsoft Purchases MessageCast
buyer=Microsoft, company=Medstory Inc.	Microsoft Announces Planned Acquisition of Medstory, Inc.
buyer=Microsoft, company=Medstory	Microsoft has acquired Medstory, a vertical search engine for health information.
buyer=Microsoft, company=Medstory	Microsoft has bought Medstory because it ' s an" intelligent" and" intuitive search technology".
buyer=Microsoft, company=Medstory	Microsoft Announces Acquisition of Medstory
buyer=Microsoft, company=Medstory	Microsoft acquired Medstory, a healthcare search company and added the company to its healthca...
buyer=Microsoft, company=media\/video sharing ...	Microsoft has bought the assets of media/video sharing service WebFives, for an undisclosed amo...
buyer=Microsoft, company=Media Sharing Servic...	Microsoft Buys Assets of Media Sharing Service WebFives
buyer=Microsoft, company=master-data manage...	Microsoft on Thursday said it has acquired master-data management vendor Stratature, and plans ...
buyer=Microsoft, company=Master Data Manage...	Microsoft Buys Master Data Management Vendor Stratature
buyer=Microsoft, company=Massive Inc.	Microsoft Bought Massive Inc., Now What?
buyer=Microsoft, company=Massive Inc.	Microsoft acquired Massive Inc. coming on two years ago

Figure 45 : Résultat brut de l'extraction d'information, sans regroupement des résultats

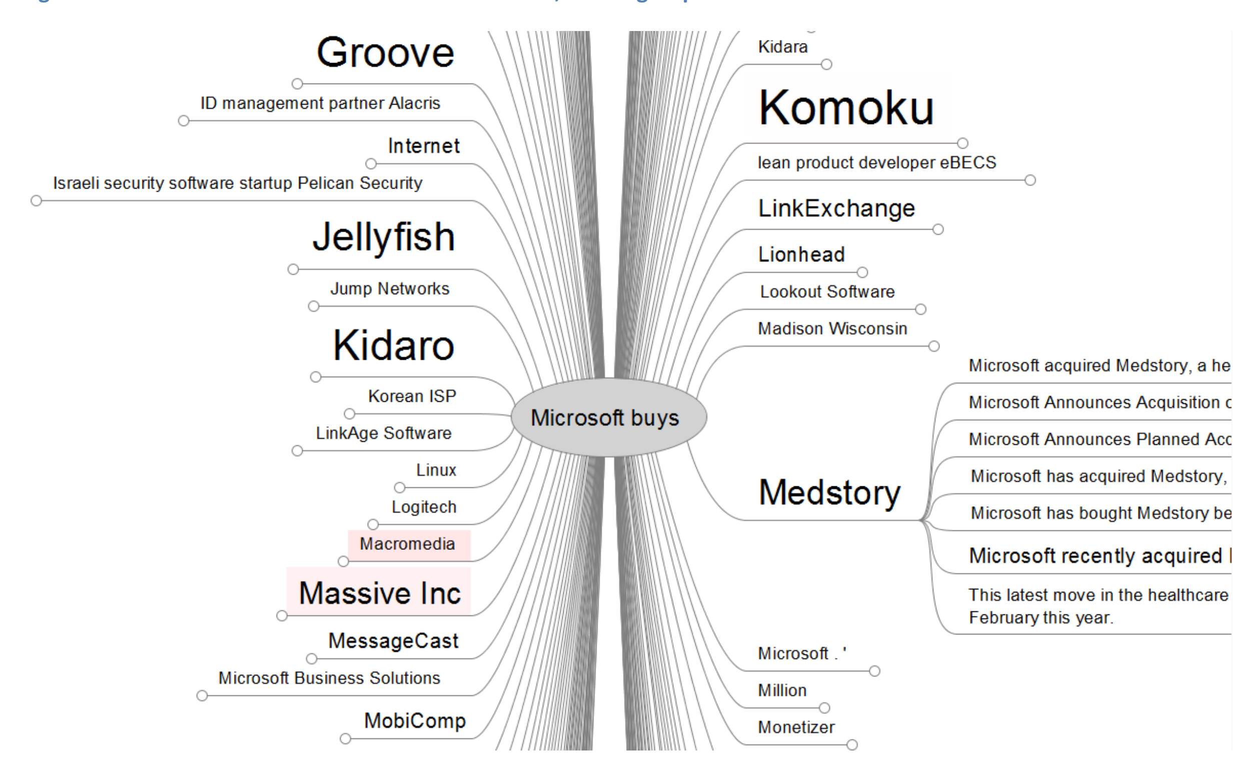


Figure 46 : Visualisation de l'extraction d'information avec regroupement des résultats

Le regroupement des résultats permet de les rassembler et de les afficher d'une façon plus synthétique : ici, on obtient 245 sociétés distinctes, après fusion des doublons. Le résultat peut être exporté sous forme de carte heuristique –ou *mind map* en anglais– (voir figure 46) pour en améliorer la lisibilité. La taille de la police de caractère servant à afficher les noms de société y est proportionnelle au nombre de documents où l'information a été trouvée.

C. Acquisition de connaissances spécifiques à un domaine applicatif

1. Objectif

L'apprentissage des spécificités d'un nouveau domaine est souvent le point faible des solutions d'analyse sémantique. En effet, cette étape peut nécessiter des semaines, voire des mois, de travail humain. Quand on analyse des avis de consommateurs, cette étape est cruciale. En effet, on ne s'exprime pas de la même façon, ni sur le fond ni sur la forme, quand on écrit un avis sur sa banque ou sur l'enseigne où l'on a fait ses courses. Les produits et services peuvent avoir des caractéristiques très variables : par exemple, l'humidité peut être un attribut positif (si on parle d'un humidificateur d'air) ou négatif (si c'est une tente de camping). Nous commencerons donc par présenter ici notre démarche d'acquisition des connaissances spécifiques à un domaine particulier.

Nous avons présenté en IV.E (page 86) notre vision d'une approche pragmatique consistant à mixer, d'une façon semi-automatique, une démarche descendante (modélisation du monde *a priori*) avec une démarche ascendante (exploitation du corpus à traiter) pour disposer du lexique le mieux adapté à une application donnée. Dit autrement, notre approche cherche à constituer rapidement l'extension spécifique nécessaire pour enrichir le lexique sémantique standard d'Antelope et l'adapter à une application donnée. Nous en illustrerons concrètement l'intérêt dans deux cas : l'analyse d'avis de consommateurs[174] d'une part (chapitre D) et de documents RH d'autre part (chapitre E).

2. Collecte d'un corpus relatif au domaine considéré

La découverte d'un domaine que l'on souhaite analyser nécessite de la « matière première ». Une étape préliminaire est donc la collecte d'un corpus de documents relatifs à ce secteur. Le cas le plus simple se présente quand un corpus représentatif peut être fourni en début de projet ; mais ce n'est pas toujours le cas.

Quand un tel corpus n'est pas disponible, nous commençons par collecter sur le Web public quelques milliers de documents issus de multiples canaux[175], de façon à avoir de la diversité. Quand nous disposons d'un début d'ontologie du domaine (notamment sur la partie produits), nous pouvons piloter un moteur de recherche du Web pour automatiser cette collecte. Nous appliquons ensuite notre démarche d'acquisition de connaissances.

3. Découverte des termes du domaine

Une étape d'extraction terminologique fait émerger les termes les plus fréquents du corpus collecté : mots simples (« banque », « compte »…) ou expressions multi-mots (« livret A », « chargé de clientèle », « banque privée » …). Une procédure interactive permet d'enlever les termes jugés inappropriés. Le reste constitue un ensemble de termes pertinents.

---

[174] L'équipe Proxem a appliqué cette approche avec succès sur des secteurs très différents, notamment la grande distribution, la banque de détail, l'industrie du vin, la cosmétique et l'automobile.

[175] Typiquement des sites d'avis de consommateurs (notamment www.ciao.fr), des blogs et forums, et éventuellement des sites d'actualité.

La figure 47 montre le résultat de l'extraction terminologique effectuée sur 3 500 avis publics donnés par des consommateurs sur leur banque (source : site www.ciao.fr). Cette visualisation sous forme de nuage de mots met en évidence :

- Des **banques** : BNP Paribas, Société Générale, Crédit Foncier, Banque Postale, Crédit Agricole, ING Direct, Banque Populaire, Caisse d'Epargne, autres banques.
- Des **produits** et **services** : site internet, livret A, carte bancaire, carte bleue, prêt immobilier, assurance vie.
- Des **caractéristiques** de ces produits et services : taux fixe, taux variable, débit immédiat.
- Des **opérations** courantes : ouverture de compte.
- Des **délais** : 3 semaines, 3 mois, 6 mois, 5 ans, 25 ans.



Figure 47 : Extraction terminologique de 3 500 avis publics de consommateurs sur leur banque

Nous organisons ensuite ces termes en classes sémantiques. Cette étape est actuellement manuelle. Nous commençons à essayer d'automatiser cette étape avec des approches d'analyse distributionnelle comme celle décrite dans (Van de Cruys, 2010), mais n'avons pas réussi pour l'instant à le faire d'une façon satisfaisante. Le regroupement des termes sous forme de concepts se fait en créant des relations :

- De synonymie, pour créer ou enrichir des synsets : « prêt immobilier » est synonyme de « crédit immobilier », par exemple.
- D'hyperonymie/hyponymie : « carte bleue » est un type de « carte bancaire », « taux fixe » et « taux variable » sont des « taux », etc.

4. Extension du lexique sémantique standard

Le lexique sémantique standard d'Antelope compte aujourd'hui 73 000 lemmes en français. 44 200 proviennent de WOLF ; 28 800 nouveaux lemmes ont été ajoutés au fil des projets, avec une technique qui importe des concepts par lots à partir de la Wikipédia en français, avant une validation manuelle. Les concepts spécifiques au domaine extraits à l'étape précédente viennent l'enrichir lors de chaque projet. La figure 48 montre par exemple une définition courte du concept « banque commerciale » avec comme instances hyponymes une liste de banques françaises.

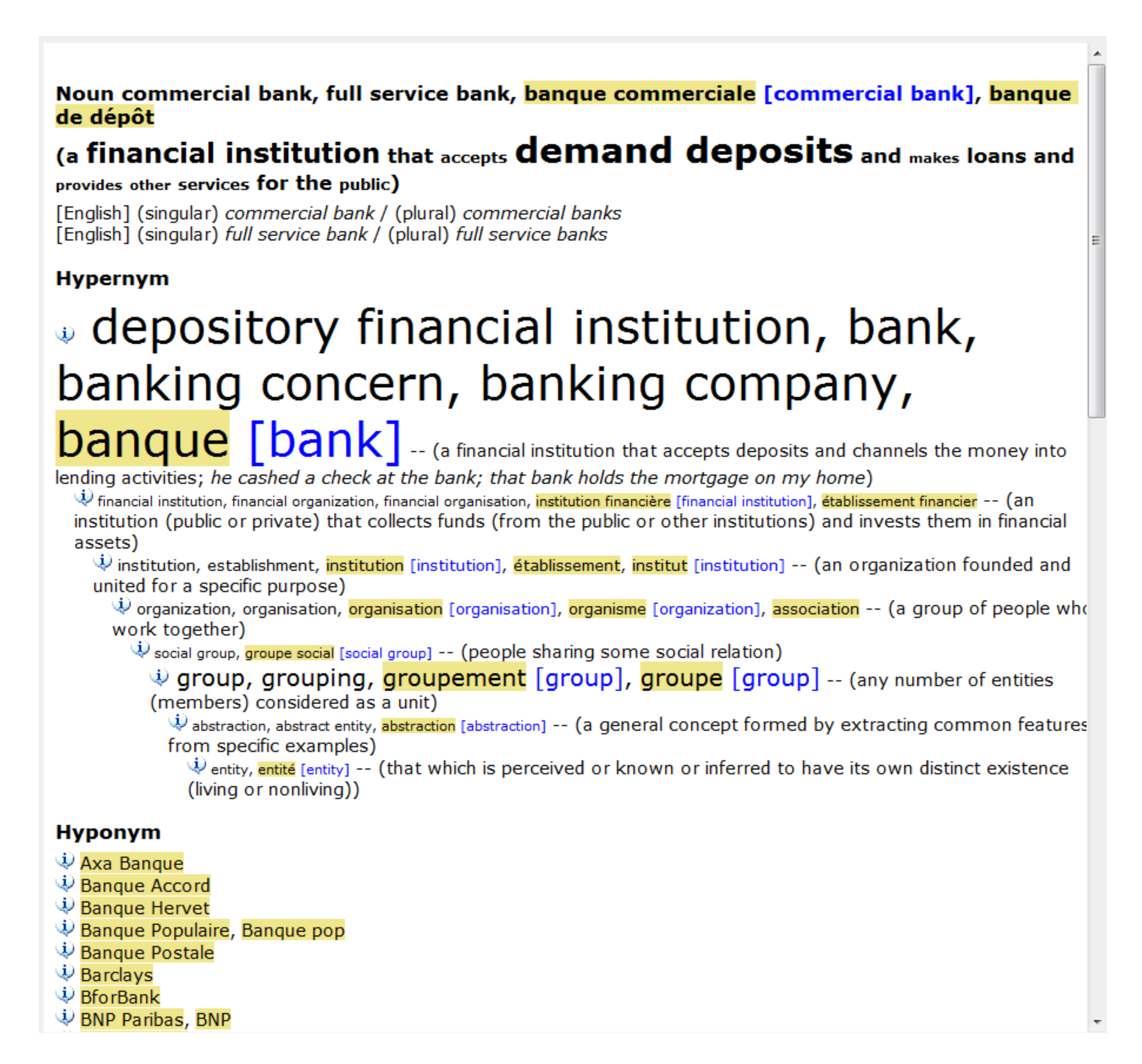


Figure 48 : Concept « banque commerciale » dans le lexique sémantique d'Antelope

5. Reconnaissance d'entités nommées du domaine

a) Amorce en utilisant des gazettes

En fonction du type d'application à réaliser, on souhaite reconnaître des entités nommées de natures très variables. La veille économique vise classiquement à identifier des personnes, lieux et organisations. Dans un contexte d'enseigne de grande distribution, les entités intéressantes à détecter sont plutôt les produits, marques et concurrents cités, ainsi que des concepts liés au métier (risque sanitaire ou risque juridique, par exemple). Dans le domaine des ressources humaines, on cherchera à extraire les métiers, compétences, diplômes, langues, etc.

Le lexique sémantique d'Antelope permet de créer des gazettes (Cf. V.B.2, page 95). En cas d'ambiguïté possible, les termes des gazettes sont associés à des mots clés activateurs ou inhibiteurs[176]. Une première passe de reconnaissance d'entités nommées peut alors être effectuée sur le corpus grâce à ces gazettes. Elle produit un corpus où les entités nommées du domaine sont annotées. La figure 49 présente quelques phrases tirées d'avis de consommateurs, où des marques apparaissent en rouge (Garnier…) et des produits en bleu (merguez, sommier, matelas…).

[176] Par exemple, le sens ORANGE#1$_{\text{[fruit]}}$ sera associé aux mots clés activateurs jus, fruit, pulpe… De même, ORANGE#2$_{\text{[marque télécom]}}$ sera lié à internet, ADSL, contrat, carte SIM, opérateur…

On a appelé la cliente pour venir chercher une commande : Lait Après Soleil de Garnier.
Acheter merguez le 27/08/2010, pas appétissant sur le barbecue de marque PRIM' GRILL
CDE larroche mazet Cabernet 2 cartons
J'ai reçu le sommier et matelas de marque ivana
Bonjour, je voulais commander un automatisme pour portail à mon magasin de Béthune

Figure 49 : Reconnaissance initiale d'entités nommées par gazettes

Les gazettes que nous utilisons peuvent s'avérer incomplètes, soit parce que les sources qui ont permis leur création ne sont pas exhaustives, soit parce que de nouveaux concepts apparaissent au fil du temps. Afin de contourner ce problème, nous tentons de détecter automatiquement sur corpus des nouveaux concepts, qui seront soumis à une validation humaine ; un linguiste peut alors valider ou infirmer les nouveaux concepts proposés. Nous utilisons pour cela différents mécanismes d'apprentissage ; nous en présentons deux ici, le premier utilisant les CRF, et le second qui opère par exploration des énumérations.

b) Généralisation par apprentissage utilisant les CRF

Un apprentissage utilisant le mécanisme des CRF (Cf. V.B.3, page 97) est alors effectué sur ce corpus annoté. La figure 50 montre (en souligné) les résultats de cet apprentissage, qui propose de nouvelles entités nommées ou complète celles qui étaient déjà identifiées.

Par exemple, de nouvelles marques sont détectées. Le système a appris que le mot (ou groupe de mot) qui suit le terme « marque » en est justement une. Il détecte alors les marques « ivana » et « PRIM' GRILL » qui n'existaient pas dans les gazettes ; de même, il considère que le nom de la marque est « Laroche Mazet » (et non « Laroche »).

En ce qui concerne les produits, le système identifie correctement « lait après soleil » (et non « lait » tout seul). Il trouve aussi le produit « automatisme pour portail », inconnu jusqu'alors ; le système a donc aussi appris que, dans ce corpus, le groupe nominal qui suit le verbe « commander » est probablement un produit.

On a appelé la cliente pour venir chercher une commande : Lait <u>Après</u> <u>Soleil</u> de Garnier.
Acheter merguez le 27/08/2010, pas appétissant sur le barbecue de marque <u>PRIM'</u> <u>GRILL</u>
CDE larroche <u>mazet</u> Cabernet 2 cartons
J'ai reçu le sommier et matelas de marque <u>ivana</u>
Bonjour, je voulais commander un <u>automatisme</u> <u>pour</u> <u>portail</u> à mon magasin de Béthune

Figure 50 : Reconnaissance d'entités nommées après généralisation par apprentissage

c) Généralisation par exploration des énumérations

Une des méthodes que nous avons adoptées consiste à repérer dans le texte des énumérations de syntagmes nominaux. L'idée est que si la plupart de ces syntagmes sont des entités connues, alors les syntagmes restants sont vraisemblablement également des entités. Il y a également une forte probabilité pour que le type d'une entité inconnue soit le même que celui des entités connues qui l'entourent, ou un type proche.

Cette méthode s'est avérée très efficace pour l'adaptation des analyseurs au domaine des ressources humaines (voir plus loin la section VI.E.2). En effet, lors de l'analyse de CV ou d'offres d'emploi, on rencontre très fréquemment des listes de compétences ou de diplômes.

L'algorithme est le suivant. On part d'un texte préalablement annoté en entités nommées. La première étape consiste à explorer les groupes nominaux contigus séparés par des virgules ou des conjonctions (telles que « et » et « ou ») ; on vérifie si une entité nommée a été reconnue dans chaque groupe nominal. On examine ensuite les énumérations ayant une longueur d'au moins trois groupes nominaux, avec au moins deux groupes nominaux reconnus en tant qu'entités nommées, et au moins un groupe nominal non reconnu en tant qu'entité nommée. Le système peut alors faire l'hypothèse qu'un groupe nominal non reconnu est un bon candidat pour devenir une nouvelle entité nommée.

Il reste à associer un type au nouveau candidat, avec deux cas de figure.

- Si les autres entités reconnues dans l'énumération partagent le même type, alors ce dernier convient pour le nouveau candidat. Soit la phrase suivante, tirée d'une offre d'emploi :

  Véritable référent technique, vous êtes expert des technologies CISCO (vous faites état de certifications: CCNA, CCNP, CCSP, CCIE...).

  La reconnaissance initiale d'entités nommées avait reconnu CCSP et CCIE (soulignées en rouge) comme étant des instances de CERTIFICATION INFORMATIQUE. Le détecteur d'énumérations a trouvé une énumération de quatre groupes nominaux dont deux connus ; il peut donc proposer de typer aussi CCNA et CCNP en tant que CERTIFICATION INFORMATIQUE.

- En revanche, si les entités reconnues ont des types différents, on attribue à l'entité nommée candidate leur plus proche ancêtre commun dans la hiérarchie des types. Prenons l'exemple :

  Habitué à travailler avec l'outil informatique notamment AutoCAD, FreeCAD, MS Project.

  La reconnaissance d'entités nommées a détecté AutoCAD comme un LOGICIEL DE CAO, et MS Project comme un LOGICIEL DE GESTION DE PROJET. Le système peut donc inférer que FreeCAD est aussi une entité nommée candidate. Pour déterminer son type, on cherche le plus proche ancêtre commun de LOGICIEL DE GESTION DE PROJET et LOGICIEL DE CAO, qui sont tous deux des hyponymes de LOGICIEL. Le système peut proposer LOGICIEL comme type pour FreeCAD. Ensuite, lors de la validation humaine, le linguiste pourra éventuellement requalifier plus finement FreeCAD en tant que LOGICIEL DE CAO.

6. Amorce d'un plan de classification

Pour finir, un regroupement automatique des documents du corpus permet d'amorcer une proposition de plan de classification thématique. Il permet de faire apparaître des groupes de documents qui correspondent par exemple à des demandes d'informations, des félicitations ou encore aux principaux motifs d'insatisfaction.

D. Analyse d'avis de consommateurs (Ubiq)

1. Objectif

Être à l'écoute de la « voix des clients » et gérer sa e-réputation[177] sont des enjeux majeurs pour toutes les marques B2C[178]. Ubiq est une solution d'aide à la décision dédiée aux marques de la grande distribution, du commerce électronique, des cosmétiques, de la banque de détail ou de la banque en ligne (Chaumartin, 2011). Son objectif est de permettre d'identifier les attentes des consommateurs en les classant par thématiques, de détecter les tendances, d'analyser les opinions, d'anticiper les problèmes et de visualiser d'un coup d'œil les « sujets chauds » du moment.

2. Filtrage de documents provenant de sources diverses

Ubiq permet aux entreprises d'analyser les avis de consommateurs quelle qu'en soit l'origine, comme le résume le tableau 20. Les verbatims d'expression spontanée se trouvent en effet le plus souvent dispersés sur des sources externes (blogs, forums, news, RSS, tweets…) et/ou internes (mails envoyés spontanément, réponses aux questions ouvertes d'enquêtes).

Documents	Ecrits spontanément par les clients	Sollicités par la marque
Sources privées	**Voix du Client** Emails envoyés spontanément Retranscriptions téléphoniques Réclamations, courriers scannés…	**Etudes** Sondages d'opinion Enquêtes de satisfaction Evaluation d'applications mobiles…
Sources publiques	**e-réputation** Blogs, forums, réseaux sociaux Twitter, Facebook Commentaires sur les news…	**Enquête publique** Exemple : la grande enquête de Radio-France « Quel travail voulons-nous ? »

Tableau 20 : Typologie des sources traitées par Ubiq

a) Collecte d'avis sur le Web

L'explosion du Web 2.0 permet à tout-un-chacun de s'exprimer sur sa page Facebook ou sur son blog, sur des forums, sur Twitter et plus récemment sur Google+, en attendant le prochain réseau social à la mode. L'ensemble de ces contributions crée un énorme volume d'avis et de jugements plus ou moins pertinents, portant sur des personnalités, des événements ou des produits.

Nous n'entrerons pas ici dans le détail de la collecte sur le Web, qui n'est pas un problème de TAL. Les difficultés sont néanmoins nombreuses : trouver les bonnes sources, découper correctement une page Web (Cf. III.I.1), s'assurer de la récence de l'information qui s'y trouve… A ce stade, les composants d'analyse sémantique entrent en jeu, en permettant par exemple de déterminer la pertinence d'une information puis d'extraire les entités nommées.

La collecte de conversations ciblant une marque ou un produit donné sur le Web soulève des difficultés spécifiques. Séparer en amont le bon grain de l'ivraie nécessite d'appliquer des filtres de désambiguïsation, de récence et de pertinence, pour éviter de traiter des milliers de conversations trop anciennes ou sans intérêt.

[177] Ou réputation en ligne, ou notoriété numérique : l'opinion globale qu'ont les internautes sur la marque.
[178] Business to Consumer (B2C ou B to C) c'est-à-dire les activités ayant le consommateur final comme client.

Illustrons ce point à travers quelques exemples ciblant la société Carrefour. Cette marque est aussi un nom commun ; une collecte par mot clé renvoie donc aussi des documents qui sont hors sujet car contenant des homonymes de l'enseigne (« *L'accident de voiture a eu lieu sur le **carrefour** giratoire* » ; « *La Suisse est au **carrefour** de l'Europe* »). Des documents citant un magasin de l'enseigne peuvent présenter un intérêt faible (« *Le hold-up a eu lieu près du **Carrefour** de Trifouilly-les-Pâquerettes* ») ou variable selon les attentes du destinataire de l'analyse (« *Le son d'avoine, j'en ai trouvé chez **Carrefour*** » ; « *Je trouve les prix bien plus intéressants chez **Carrefour** que chez Leclerc* »).

3. Analyses effectuées par Ubiq

Les traitements sémantiques réalisés par Ubiq permettent d'extraire et d'associer aux documents de nombreuses métadonnées : marques, produits, concurrents, opinions... selon le processus présenté en figure 51.

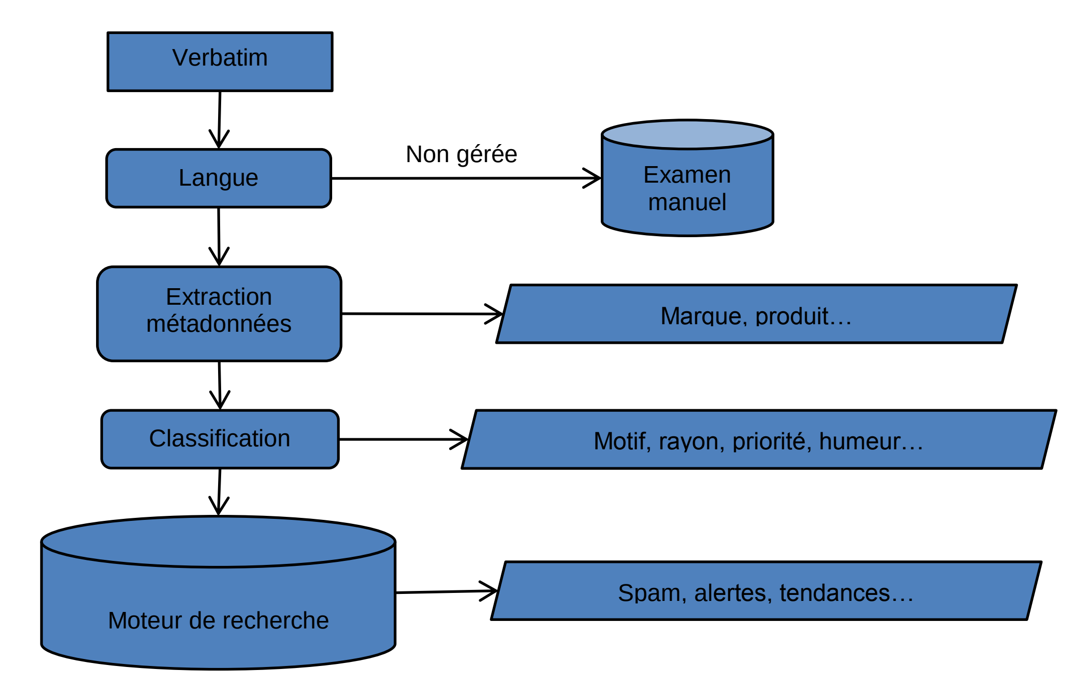


Figure 51 : Processus de l'analyse sémantique effectuée par Ubiq

a) Détection de la langue

Un premier traitement cherche à détecter la langue de chaque document. Ubiq traite ensuite les documents en écrits français ou l'anglais. Cette étape soulève son lot de difficultés. Nous faisons actuellement l'hypothèse qu'un avis est rédigé intégralement dans une langue donnée. Or, dans certain cas, il peut contenir des sections faisant alterner deux langues : un verbatim de consommateur portant sur des produits électroniques est souvent rédigé en « franglais », en mélangeant par exemple un avis exprimé en français et des citations tirées d'un manuel d'utilisation en anglais ; cela entraîne alors une détection erronée de la langue.

Nous ne traitons pas ces cas pour l'instant, car ils sont très peu fréquents dans les corpus que nous avons traités. Le problème est alors que la section de texte rédigée dans une langue détectée d'une façon incorrecte fera l'objet de traitements (comme la correction orthographique ou la reconnaissance d'entités nommées) dans la mauvaise langue, ce qui créé du bruit dans le résultat de l'analyse.

b) *Correction orthographique*

Les verbatims de consommateurs n'étant pas exempts de fautes, Ubiq procède à une étape préalable de correction orthographique. Une spécificité est d'utiliser plusieurs heuristiques pour tenir compte du contexte. Par exemple, une heuristique exploite le fait que les documents sont analysés par lots de plusieurs centaines ou plusieurs milliers. On peut alors tenir compte de la fréquence de chaque « erreur nouvelle » : si elle revient seulement 2 ou 3 fois, il doit effectivement s'agir d'une erreur ; en revanche, si des dizaines d'occurrences de la même erreur nouvelle sont présentes, il est plus probable qu'il s'agisse d'une marque ou d'un produit inconnu du lexique. Des marques nouvelles apparaissent en effet fréquemment : elles représentent souvent des mots inconnus, absents du lexique standard. Cette heuristique permet donc d'éviter qu'une marque inconnue (comme *Pet Shop* par exemple) soit improprement corrigée vers le mot connu le plus proche (vers *sex shop* dans le même exemple).

c) *Reconnaissance d'entités nommées composites*

Cette étape permet de reconnaître les principaux éléments cités dans le document : produits, enseignes, fournisseurs, concurrents, marques, thématiques, problèmes et attentes exprimés par le consommateur… Nous avons toutefois apporté des améliorations au composant standard de reconnaissance d'entités nommées, pour tirer pleinement parti du contexte.

Nous avons déjà vu comment des homonymes comme ORANGE#1$_{\text{[fruit]}}$ et ORANGE#2$_{\text{[marque télécom]}}$ sont différenciés. Ubiq sait aussi détecter les produits composés, grâce à une phase d'apprentissage sur le corpus. Nous tenons compte du fait que, sous certaines conditions[179], deux produits consécutifs n'en forment qu'un. Par exemple, « *canard à l'orange* » est correctement détecté comme un produit composé unique, rattaché au rayon des plats cuisinés, et non comme deux produits des rayons volailles (« *canard* ») et fruits & légumes (« *orange* »). Ubiq est aussi capable de reconnaître dans « *chocolat Lindt aux noisettes* » qu'on parle de « *chocolat aux noisettes* » (et que c'est un hyponyme de « *chocolat* ») de la marque Lindt.

d) *Identification des synonymes*

L'expansion des synonymes est faite lors de l'indexation sémantique, et non lors de la recherche. Cela permet de tirer pleinement partie du contexte, et d'optimiser les performances, en effectuant l'opération une fois pour toutes. Par exemple, « *serveuse* » a pour synonyme « *hôtesse* » ; le sens de « *serveur* » dépend en revanche du contexte (« *serveur informatique* » est un autre sens).

4. Exemples concrets d'analyse d'un document

a) *Contexte bancaire*

La figure 52 montre l'analyse d'un verbatim relatif au monde bancaire : les établissements, produits et opérations bancaires sont correctement identifiés.

[179] Attention aux exceptions : si la « tarte à la noix » existe effectivement, un produit « à la noix » n'est pas forcément un produit composite ; de même, le « collier de mouton » et le « collier pour chien » ne doivent pas être confondus avec un collier du rayon bijouterie.

Banque excellente, rien à redire! (2,22)
Je possède depuis quelques temps maintenant un compte chez Boursorama et je dois dire que je suis ravie !!!!J'ai commencé en douceur en ouvrant un compte perso, puis au fur et à mesure ont suivi les comptes épargnes, actifs en quelques clics ! Maintenant, c'est mon compte principal, carte bancaire et tous les prélèvements aussi.De nombreux avantages : Tout d'abord le prix : je ne paie qu' 1,5 euros par mois, pour une assurance "boursorama protection", juste parce que j'aime etre rassurée en cas d'éventuelle fraude à la CB etc.J'ai une carte visa premier, qui ne me coûte rien.Cette banque est vraiment LA banque du 21e siècle : l'interface est géniale et on y retrouve toutes les fonctionnalités possibles. Création de comptes tiers pour virements immédiate et gratuite (même pour les virements à l'étranger, en UE en tout cas). Notification par sms pour tout achat supérieur à une certaine somme, j'en passe et des meilleures.Inconvénients : je n'en vois pas, si ce n'est que j'aurais aimé prendre mon prêt immo chez eux, mais qu'ils ne prêtent pas dans le cadre de contrats de construction. Bien dommage, ça m'aurait évité quelques cheveux blancs chez les conccurents.

Figure 52 : Capture d'écran de l'analyse d'un verbatim relatif au monde bancaire

b) *Contexte grande distribution*

La figure 53 montre l'analyse d'un verbatim relatif à la grande distribution. Les produits, marques, concurrents et concepts y sont correctement détectés. La faute d'orthographe sur « bagette » est corrigée en « baguette ». Remarquons que le système procède à une normalisation des noms de produits (« papier toilette » = « PQ ») et de marques (« saint Hubert 41 » = « ST HUBERT 41 »).

Je trouve que depuis un certain temps, vous avez augmenté les prix. Avant, j'avais l'habitude d'aller au [illegible] de [illegible]. Depuis, j'achète chez Lidl, le lait à 0.55€ le litre et le saint Hubert 41 (5.20€ le kilo soit 2.60 les 2 fois 250g et bien d'autres produits. De temps en temps, je vais chez ED (il y a de moins en moins d'ED car ils deviennent Carrefour City) à 0.60€, bien moins cher que chez vous. Mais j'ai été malade sur des produits [illegible]. Donc j'y vais le moins possible. La viande, je l'achète chez Leclerc à Pantin. Très bonne et bien moins cher que chez vous et en plus elle est tendre. Je travaille à VDF (Val de fontenay RER E) et j'ai sur le trajet entre mon travail et le RER, un grand AUCHAN val de Fontenay où j'achète mes yaourts, le Papier toilette Moltonel en paquet (Chez vous les 6 paquets en feuilles sont vendu 3.07 alors que je les achète chez Auchan à 2.77 les 12), la viande etc. Mes parents habitent dans l'Oise et j'y vais tout les 15 jours. Il y a un auchan et j'y fais presque toutes mes courses (viande, crémerie, pain, PQ ...) pour les 15 jours. Avant, je travaillé au métro Strasbourg St Denis (avant de déménager à VDF) et toute les semaines, je continue d'acheter les fruits et légumes de la semaine dans un grand magasin de fruits qui les vends pas cher. Comme vous pouvez le voir, je n'achète pas grand chose dans votre supermarché, a part la bagette de 400 g qui est délicieuse et de temps en temps le pain en tranche de la marque carrefour discount (le - cher) si je n'ai pas le temps d'aller chez Auchan ou Lidl. Je connais plusieurs personnes qui font leur courses dans différents magasins et comparent les prix.

Figure 53 : Capture d'écran de l'analyse d'un verbatim relatif à la grande distribution

5. Détection des tendances

Les traitements évoqués jusqu'ici portent sur les documents pris individuellement. Ubiq regroupe ensuite toutes les informations extraites, pour donner une vue d'ensemble sur ce qui se passe sur une période de temps et en dégager les tendances ; le composant de regroupement de documents d'Antelope est utilisé à cet effet. La figure 54 montre comment le module de détection des tendances affiche sur une *timeline* synthétique les documents de chaque semaine (période d'activité commerciale de référence dans la grande distribution). La taille de chaque regroupement est proportionnelle au nombre de documents qui la constituent ; leur couleur indique le motif principal (par exemple, les ruptures de stocks apparaissent en vert, les problèmes de qualité en rouge, etc.). L'utilisation d'une évolution du composant d'analyse de sentiments (Cf. V.D, page 115) permet de mesurer l'évolution dans le temps du nombre d'avis positifs et négatifs. Cette interface rend service au quotidien à plusieurs publics : la direction du marketing dispose d'un instantané des avis exprimés par les consommateurs au niveau national ; un directeur de magasin a une idée précise de ce qui se passe dans son périmètre ; un responsable qualité, spécialisé dans une gamme de produits, peut se faire une opinion sur l'évolution de la qualité et découvrir d'éventuels problèmes de production.

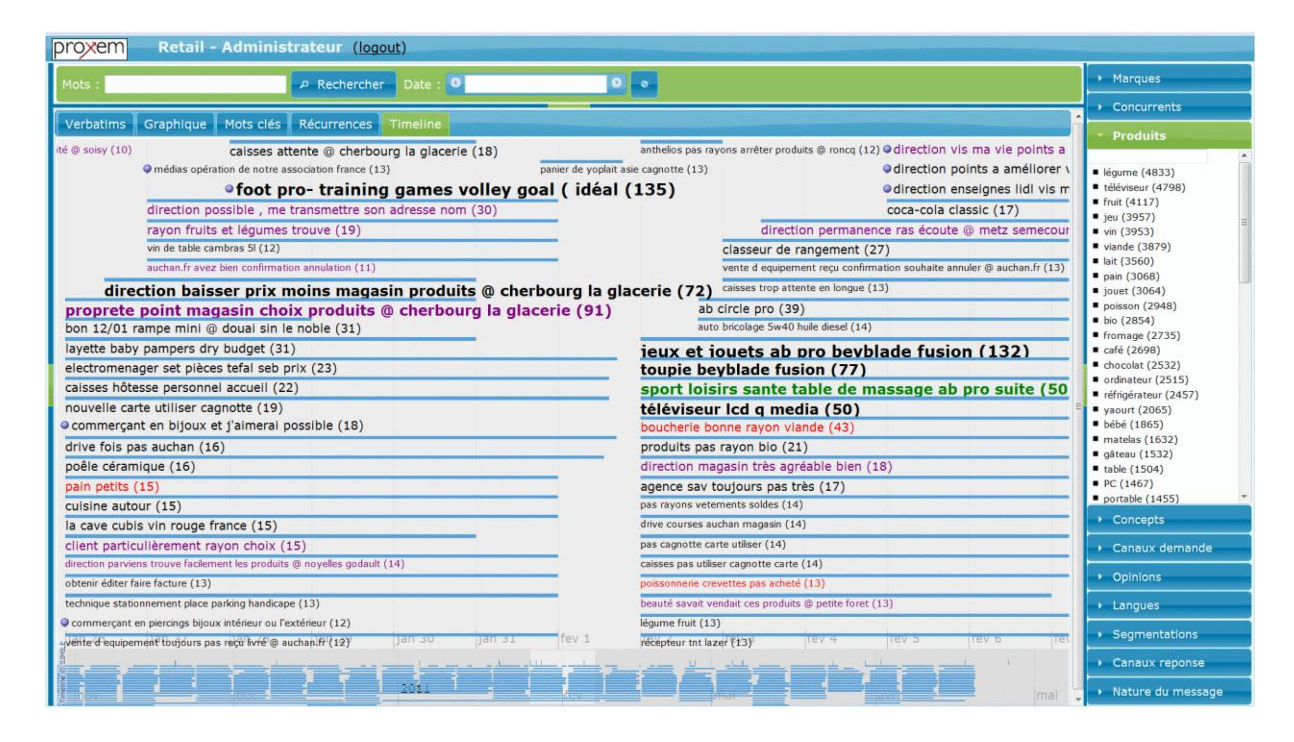


Figure 54 : Vision de synthèse de plus de 10 000 documents sur deux semaines

6. Interface de la solution Ubiq

a) Recherche et analyse multidimensionnelle



Figure 55 : Analyse multidimensionnelle permettant d'effectuer un zoom jusqu'au verbatim

Lors de la phase d'analyse, Ubiq extrait des conversations les produits (et rayons), les enseignes concurrentes, les marques, les thématiques et concepts ainsi que les problèmes exprimés par les consommateurs. Comme montré en figure 55, Ubiq permet ensuite d'effectuer simplement des recherches par mots clés, par facettes[180], ou par une combinaison simultanée de ces deux types de recherche. Ubiq permet aussi de chercher les verbatims similaires à un verbatim donné, avec ou sans limitation de plage de temps, ce qui est pratique pour déterminer si un phénomène est chronique ou ponctuel.

b) Rapports et tableaux de bord

Une fois les documents analysés, on peut facilement faire des requêtes sur les documents, mesurer l'évolution dans le temps d'un phénomène, définir des indicateurs de synthèse pour constituer un tableau de bord, effectuer des analyses croisant deux axes, etc. La figure 56 montre des exemples de tableaux de bord synthétiques générés par Ubiq, qui servent d'outil de pilotage quotidien : évolution des avis au fil du temps, griefs les plus évoqués, etc.

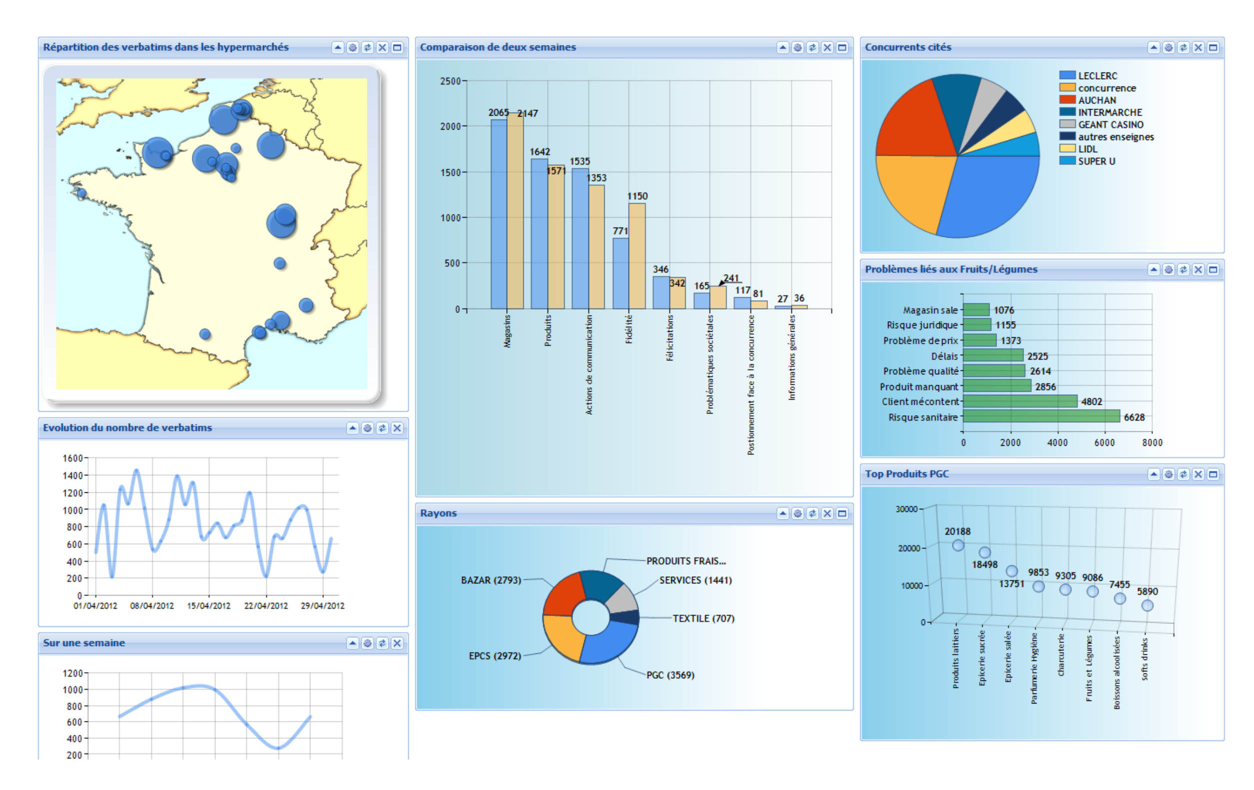


Figure 56 : Tableaux de bord synthétiques d'Ubiq

E. Analyse d'offres d'emploi et de CV (Ubiq RH)

1. Objectif

Les offres d'emploi et CV disponibles en ligne constituent un matériau potentiellement très riche, disponible en temps réel. Son analyse est rendue difficile en raison de la masse d'informations à analyser, de la variété des supports de publication en ligne, de la faible standardisation des formats

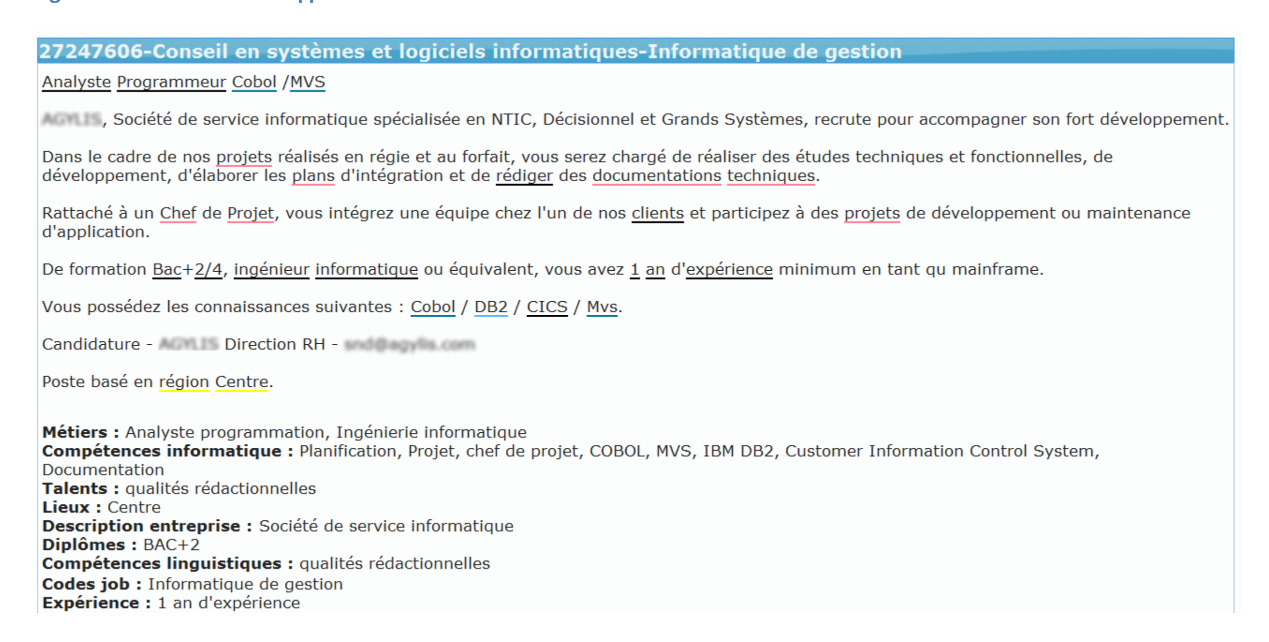
[180] Un moteur de recherche à facettes affiche des données structurées extraites du texte, en les regroupant par catégories. La figure 55 montre sur la gauche de la capture d'écran les thématiques calculées (motifs et rayons) et sur la droite les entités nommées (marques, concurrents, produits, concepts…) qui constituent les différentes facettes dans le cas d'espèce.

d’annonce, de l’absence de référence à des nomenclatures communes et du caractère souvent implicite des contenus. L’analyse sémantique offre un moyen de traiter rapidement des volumes importants de documents RH.

Après avoir été originellement conçue pour traiter les avis de consommateurs, la solution Ubiq a été adaptée au domaine RH par l’équipe Proxem pour analyser des offres d’emploi et des CV. Ubiq RH permet aussi de chercher les CV correspondant le mieux à une offre ou les meilleurs postes pour un profil donné. Un point à souligner est que le code des deux versions est quasiment identique, l’adaptation au domaine se faisant par simple paramétrage. La figure 57 montre l’interface d’Ubiq dans ce contexte : on voit à gauche les taxonomies correspondant aux métiers et aux compétences. La figure 58 détaille les informations extraites suite à l’analyse d’un CV.



Figure 57 : Interface d’Ubiq permettant la recherche dans les documents RH



27247606-Conseil en systèmes et logiciels informatiques-Informatique de gestion

Analyste Programmeur Cobol /MVS

[redacted], Société de service informatique spécialisée en NTIC, Décisionnel et Grands Systèmes, recrute pour accompagner son fort développement.

Dans le cadre de nos projets réalisés en régie et au forfait, vous serez chargé de réaliser des études techniques et fonctionnelles, de développement, d'élaborer les plans d'intégration et de rédiger des documentations techniques.

Rattaché à un Chef de Projet, vous intégrez une équipe chez l'un de nos clients et participez à des projets de développement ou maintenance d'application.

De formation Bac+2/4, ingénieur informatique ou équivalent, vous avez 1 an d'expérience minimum en tant qu mainframe.

Vous possédez les connaissances suivantes : Cobol / DB2 / CICS / Mvs.

Candidature - [redacted] Direction RH - [redacted]

Poste basé en région Centre.

Métiers : Analyste programmation, Ingénierie informatique
Compétences informatique : Planification, Projet, chef de projet, COBOL, MVS, IBM DB2, Customer Information Control System, Documentation
Talents : qualités rédactionnelles
Lieux : Centre
Description entreprise : Société de service informatique
Diplômes : BAC+2
Compétences linguistiques : qualités rédactionnelles
Codes job : Informatique de gestion
Expérience : 1 an d'expérience

Figure 58 : Un exemple d’analyse de CV, avec les différentes informations extraites

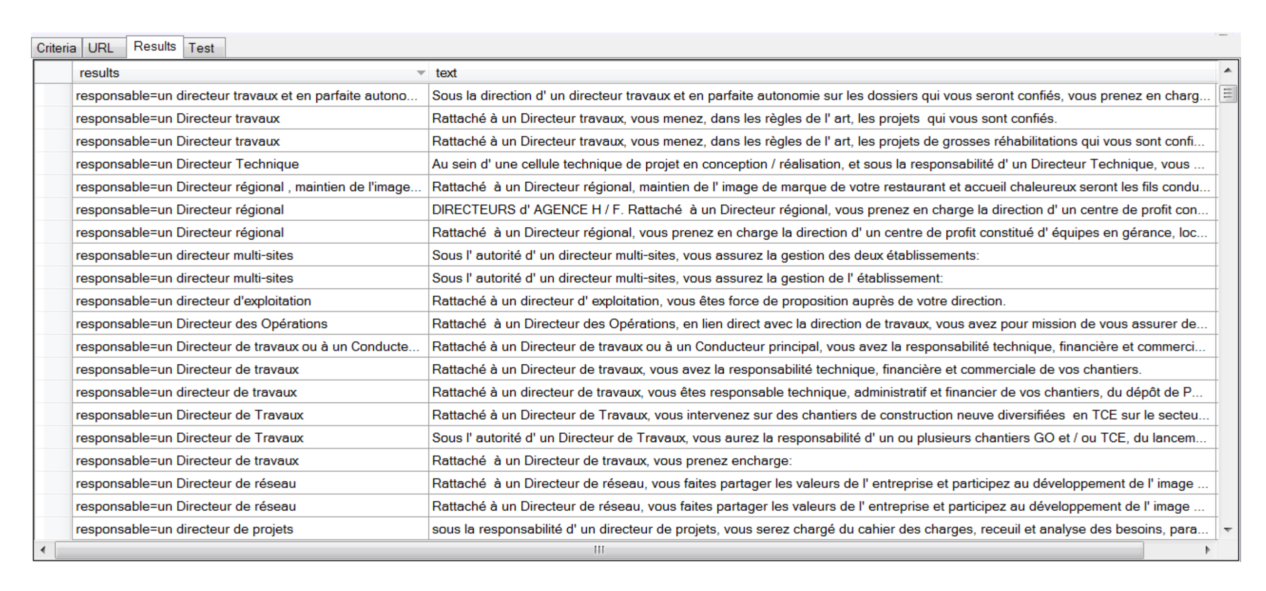
2. Adaptation d’Ubiq au domaine RH

a) Adaptation des analyseurs

Différents types d’ambiguïté sont présents dans les documents RH. Un nom propre comme Orange peut désigner une ville (où habite un candidat, où un poste est à pourvoir…), une entreprise (celle qui recrute, ou bien une expérience citée dans un CV) ou encore un patronyme. Un nom de métier désigne des réalités différentes en fonction du secteur : par exemple, on recrute des architectes en informatique et dans le BTP. Plusieurs villes portent le même nom : Evry peut être le chef-lieu du département 91 ou une autre ville du département 89. Un nombre, 50000 par exemple, désigne en fonction du contexte un code postal ou une rémunération.

La démarche d’acquisition de connaissances présentée au chapitre C a été appliquée en s’appuyant sur un corpus d’approximativement 100 000 CV et 50 000 offres. Au final, les informations extraites ici par Ubiq sont : les métiers, les compétences sous-jacentes, les talents, les expériences, les compétences linguistiques, les diplômes, les types d’entreprise, les types de poste, les secteurs, les habilitations, les éléments de rémunération ainsi que les lieux. Ces travaux d’adaptation au domaine RH ont contribué aux publications (Loth *et al.*, 2010) et (Chaumartin, 2012).

En ce qui concerne l’identification des métiers, nous avons dû prendre en compte une difficulté dans l’analyse de certaines offres. Le composant de reconnaissance d’entités nommées extrait les métiers cités, mais on peut en trouver plusieurs dans une même offre ; en effet, si le poste à pourvoir apparaît toujours explicitement (« entreprise de BTP recrute un **conducteur de travaux** »), l’offre peut aussi mentionner un rattachement hiérarchique (« sous l’autorité du **directeur régional**, vous… »). Nous avons alors mis en œuvre le composant d’extraction de relations (Cf V.C, page 106) pour gérer ce second cas[181], de façon à ne pas confondre le profil recherché avec son supérieur direct. La figure 59 montre des exemples de résultats obtenus.



results	text
responsable=un directeur travaux et en parfaite autono...	Sous la direction d' un directeur travaux et en parfaite autonomie sur les dossiers qui vous seront confiés, vous prenez en charg...
responsable=un Directeur travaux	Rattaché à un Directeur travaux, vous menez, dans les règles de l' art, les projets qui vous sont confiés.
responsable=un Directeur travaux	Rattaché à un Directeur travaux, vous menez, dans les règles de l' art, les projets de grosses réhabilitations qui vous sont confi...
responsable=un Directeur Technique	Au sein d' une cellule technique de projet en conception / réalisation, et sous la responsabilité d' un Directeur Technique, vous ...
responsable=un Directeur régional , maintien de l'image...	Rattaché à un Directeur régional, maintien de l' image de marque de votre restaurant et accueil chaleureux seront les fils condu...
responsable=un Directeur régional	DIRECTEURS d' AGENCE H / F. Rattaché à un Directeur régional, vous prenez en charge la direction d' un centre de profit con...
responsable=un Directeur régional	Rattaché à un Directeur régional, vous prenez en charge la direction d' un centre de profit constitué d' équipes en gérance, loc...
responsable=un directeur multi-sites	Sous l' autorité d' un directeur multi-sites, vous assurez la gestion des deux établissements:
responsable=un directeur multi-sites	Sous l' autorité d' un directeur multi-sites, vous assurez la gestion de l' établissement:
responsable=un directeur d'exploitation	Rattaché à un directeur d' exploitation, vous êtes force de proposition auprès de votre direction.
responsable=un Directeur des Opérations	Rattaché à un Directeur des Opérations, en lien direct avec la direction de travaux, vous avez pour mission de vous assurer de...
responsable=un Directeur de travaux ou à un Conducte...	Rattaché à un Directeur de travaux ou à un Conducteur principal, vous avez la responsabilité technique, financière et commerci...
responsable=un Directeur de travaux	Rattaché à un Directeur de travaux, vous avez la responsabilité technique, financière et commerciale de vos chantiers.
responsable=un directeur de travaux	Rattaché à un directeur de travaux, vous êtes responsable technique, administratif et financier de vos chantiers, du dépôt de P...
responsable=un Directeur de Travaux	Rattaché à un Directeur de Travaux, vous intervenez sur des chantiers de construction neuve diversifiées en TCE sur le secteu...
responsable=un Directeur de Travaux	Sous l' autorité d' un Directeur de Travaux, vous aurez la responsabilité d' un ou plusieurs chantiers GO et / ou TCE, du lancem...
responsable=un Directeur de travaux	Rattaché à un Directeur de travaux, vous prenez encharge:
responsable=un Directeur de réseau	Rattaché à un Directeur de réseau, vous faites partager les valeurs de l' entreprise et participez au développement de l' image ...
responsable=un Directeur de réseau	Rattaché à un Directeur de réseau, vous faites partager les valeurs de l' entreprise et participez au développement de l' image ...
responsable=un directeur de projets	sous la responsabilité d' un directeur de projets, vous serez chargé du cahier des charges, receuil et analyse des besoins, para...

Figure 59 : Exemples de détection de rattachement hiérarchique dans des offres d’emploi

[181] Avec des patrons morphosyntaxiques comme :

- sous la (responsabilité | conduite | direction | coordination | autorité) du respX:anyNoun
- rattaché [directement] au respX:anyNoun

Pour finir sur l'adaptation des analyseurs au domaine RH, rappelons que la méthode d'apprentissage par exploration des énumérations a donné ici de bons résultats. En effet, les offres et les CV présentent souvent des énumérations (comme les listes de compétences ou de diplômes).

b) Etude de référentiels existants

Notre objectif était de disposer d'un référentiel complet sur les métiers et les compétences sous-jacentes. Nous avons étudié certaines nomenclatures des métiers utilisées dans le monde professionnel. Certaines sont internationales (ISCO-08 – *International Standard Classification of Occupations*[182] du Bureau International du Travail), d'autres nationales (ROME – *Répertoire Opérationnel des Métiers et des Emplois*[183] du Pôle Emploi ; annuaire des métiers[184] de l'APEC ; FAP-2009 – *nomenclature des Familles Professionnelles*[185] de l'INSEE). Cette étude nous a permis de constater que ces nomenclatures officielles intègrent rarement d'une façon directe les compétences recherchées dans les offres d'emploi ou citées dans les CV. Cela a conduit l'équipe Proxem, lors de la phase d'adaptation des analyseurs, à acquérir les compétences et talents associés à un métier donné avec une approche semi-supervisée.

3. Similarité entre documents exploitant la taxonomie

Grâce aux informations extraites lors de l'analyse sémantique, Ubiq dispose d'une connaissance fine des métiers et des compétences sous-jacentes. Cela permet de trouver les meilleurs profils correspondant à une offre ou, d'une façon symétrique, les annonces correspondant le mieux à un CV donné.

Une partie importante de l'analyse sémantique d'un document RH consiste à reconnaître des entités nommées à l'intérieur d'un document. Elles sont organisées en arborescence et forment une taxonomie du domaine RH. L'exploitation de cette taxonomie, avec la prise en compte de la distance entre deux concepts, permet d'améliorer sensiblement la pertinence des résultats lors des recherches effectuées par les candidats ou les recruteurs.

Par exemple, un CV qui contient une seule fois le mot « Java », mais plusieurs termes comme « JSP », « Struts » et « Hibernate » (technologies liées à Java) sera correctement identifié comme étant celui d'un profil expérimenté dans le domaine Java. De même, dans le domaine informatique, une compétence .NET est (toutes choses égales par ailleurs) plus proche d'une compétence Java qu'une compétence COBOL.

F. Autres projets de R&D utilisant Antelope

1. Une plate-forme utilisable… et utilisée !

Nous souhaitons conclure cette partie en montrant que la plate-forme Antelope est utilisable –et effectivement utilisée– non seulement par l'équipe Proxem, mais aussi par des internautes n'ayant eu aucun échange direct avec Proxem. A nos yeux, c'est un point essentiel pour juger de la maturité d'une plate-forme. Un prérequis est évidemment de disposer d'un minimum d'éléments qui en

---

[182] http://www.ilo.org/public/english/bureau/stat/isco/isco08/index.htm
[183] http://www.pole-emploi.fr/candidat/les-fiches-metiers-@/index.jspz?id=681
[184] http://annuaire-metiers.jd.apec.fr/
[185] http://www.travail-emploi-sante.gouv.fr/IMG/pdf/FAP-2009_Introduction_et_table_de_correspondance.pdf

facilitent la prise en main : Antelope est utilisable sans contrainte pour la recherche et l'enseignement, et dispose d'un programme d'installation, d'une documentation d'utilisation de 65 pages, d'un fichier d'aide et de deux programmes de démonstration (dont l'application de veille économique présentée au chapitre B) qui recompilent sans difficulté particulière.

A ce jour, plus de 2 500 internautes ont téléchargé Antelope et l'ont testée *out-of-the-box*. Elle a été utilisée dans le cadre de projets industriels, par exemple pour créer des agents conversationnels. Certains internautes ont contribué à la plate-forme par leurs remarques. D'autres ont évalué ou mis en œuvre la plate-forme pour des projets de recherche ; en voici une bibliographie commentée.

2. Bibliographie commentée de publications mentionnant Antelope

Voici une liste non exhaustive de publications qui citent Antelope. Deux points nous semblent intéressants à souligner :

- D'une part, le choix d'Antelope est souvent justifié par la facilité de mise en œuvre conjointe de plusieurs composants (par exemple le lexique sémantique et un analyseur syntaxique). En cela, nous pensons avoir atteint notre objectif d'intégration.
- D'autre part, si la moitié de ces articles concernent principalement le TAL, les autres présentent des applications concrètes dans d'autres domaines : domotique, e-learning, génie logiciel, gestion de crises ou formation professionnelle. Antelope aura donc contribué à une démocratisation du TAL et à son ouverture vers d'autres domaines.

(Varga *et al.*, 2010) montre comment extraire, grâce à ConceptNet (Cf. IV.D.4.b) et Antelope, le sujet principal d'un document écrit en anglais ainsi que les concepts clés qu'il contient. Ces informations sont ensuite présentées à l'utilisateur d'une façon interactive, sous forme d'un nuage de concepts.

(Soto *et al.*, 2009) explore l'utilisation d'ontologies comme WordNet et YAGO (Cf. IV.A.2.d) en tant que bases de connaissances, afin de construire automatiquement les objets d'apprentissage pour des applications d'e-learning. Le système présenté génère des exercices interactifs au format HTML pour différents cours. Sur la base d'un exercice précédemment écrit manuellement et des ontologies précitées, le système permet de créer de nouvelles versions de l'exercice et d'en changer le contenu spécifique. Le lexique sémantique d'Antelope est utilisé pour la génération des exercices.

(van Willegen *et al.*, 2009) investigue la similarité entre mots et définit une distance d'affinité sémantique. Cette distance utilise le lexique sémantique d'Antelope (notamment WordNet) où deux concepts peuvent être reliés par une chaîne de synonymes, le nombre de « sauts » correspondant à la distance entre les mots.

(Despotakis, 2011) rappelle que les *serious games* ont une importance grandissante en formation professionnelle ; les apprenants améliorent leurs compétences en étant immergés dans des simulations d'activités réelles. La personnalisation de l'expérience et l'adaptation aux besoins de l'apprenant jouent un rôle clé dans l'utilisabilité de ces environnements. L'article présente le cas des formations aux entretiens d'embauche, avec une méthode pour les personnaliser grâce à des textes collectés sur les réseaux sociaux. Antelope y est utilisée pour effectuer l'analyse sémantique de commentaires d'internautes sur des vidéos de recrutement. Des thématiques proches sont explorées dans (Ammari *et al.*, 2011)

(Rouillard, Tarby, 2011) présente plusieurs solutions (voix, geste, interface haptique) pour communiquer avec un environnement domotique. Des solutions à large couverture de vocabulaire sont maintenant disponibles pour la reconnaissance vocale. L'article présente une architecture de traitement qui enchaîne plusieurs modules externes. Antelope est le candidat proposé pour effectuer le traitement sémantique en sortie du module de reconnaissance vocale. Il est intéressant pour un système domotique de savoir qu'un FOUR A MICRO-ONDES est une sorte d'APPAREIL ELECTRIQUE, par exemple ; le lexique sémantique permet donc d'enrichir le dialogue entre la maison et ses habitants.

D'après (Doumit, Minai, 2011), les médias ont un biais. Un article politique peut par exemple présenter une influence libérale, conservatrice ou centriste. Des recherches récentes visent à identifier et classer ces biais grâce à l'analyse des sentiments exprimés par les adjectifs et adverbes trouvés dans les articles de presse ; mais les méthodes d'évaluation des modèles utilisés les rendent critiquables. L'article propose un système d'extraction d'information et d'analyse d'articles politiques. L'idée est de combiner l'allocation de Dirichlet latente (LDA) et des techniques de TAL (avec Antelope pour l'analyse de la structure sémantique des articles) pour identifier les « traits de personnalité » spécifiques d'un média par rapport à différents sujets.

(Ferreira, da Silva, 2008) souligne que nombre de projets informatiques échouent, du fait de spécifications ambiguës ou d'exigences incohérentes. L'article propose une nouvelle approche socio-technique pour surmonter ces problèmes de qualité des logiciels. Il met en avant l'intérêt de disposer d'une plate-forme qui favorise l'implication des parties prenantes pour capturer leurs besoins implicites, et permette l'application de bonnes pratiques de génie logiciel. L'article propose une approche pour améliorer la qualité et la rigueur des spécifications en combinant les techniques du Web 2.0 et des outils de TAL (dont Antelope) pour aider à la validation des exigences.

(Fitrianie *et al.*, 2010) rappelle qu'une gestion de crise implique une collaboration entre de nombreux interlocuteurs. Pour coordonner leurs activités, ils doivent s'appuyer sur des informations détaillées et précises sur la crise et son environnement. Pour assurer la collaboration des services d'urgence et apporter rapidement des soins aux victimes, il est nécessaire de fournir une vue d'ensemble avec des informations mises à jour en permanence. Or les approches actuelles pour construire une telle vue se heurtent à plusieurs difficultés : événements en constante évolution, informations réparties entre des sites géographiques éloignés, difficulté de vérification des informations obtenues… Antelope peut servir comme composant d'analyse textuelle et d'extraction d'information des flux temps réel.

Partie VII. Interface syntaxe-sémantique

A. Introduction

1. Premier bilan sur les objectifs visés

Il est temps de dresser un premier bilan sur les objectifs que nous avons atteints (ou non) dans le cadre du développement de la plate-forme. Commençons par le point qui nous semble satisfaisant. Notre premier objectif était de rendre les ressources génériques et interchangeables pour une tâche donnée, de façon à accélérer et simplifier le développement d'applications de TAL. Cet objectif, qui était loin d'être acquis au démarrage de nos travaux de recherche, nous semble aujourd'hui réalisé. Il a nécessité quelques années de travail personnel et un important effort d'ingénierie avec l'appui d'une équipe.

La partie VI a présenté des exemples concrets d'applications construites en assemblant des composants d'Antelope. Nous y avons montré comment une approche semi-supervisée de l'acquisition de connaissances spécifiques à un domaine permet une industrialisation des techniques d'extraction d'information. Leur mise en œuvre permet, à partir de textes tout-venant issus de corpus spécialisés, de produire une représentation du sens plus riche que celle manipulée par un moteur de recherche classique (Cf. II.A.3, page 10). En effet, une partie significative des éléments est désambiguïsée grâce à la reconnaissance d'entités nommées et la structure reliant ces éléments est partiellement préservée avec l'extraction de relations. Nous sommes en présence d'une forme restreinte d'ISS, qui constitue un progrès par rapport au vecteur termes-fréquences où le sens est « aplati » avec une compression destructive de l'information.

Notre objectif à long terme reste de rendre le texte calculable, après une éventuelle adaptation à un domaine particulier, en développant une ISS générale, capable de produire la représentation sémantique « idéale » évoquée au chapitre II.C (page 15) : un graphe hiérarchisé de relations prédicat-argument entre des acceptions lexicales désambiguïsées. La réalisation d'une telle ISS est donc subordonnée à une désambiguïsation fine des différents éléments langagiers. A notre connaissance, cet objectif n'est encore atteint par aucun système aujourd'hui.

2. Plan de cette partie

Notre système n'est pas encore à la hauteur de nos ambitions, mais nous avons plusieurs développements en cours que nous souhaitons présenter dans cette partie. Nous y exposerons nos idées sur la prise en compte de l'ambiguïté et sur l'écriture d'un prototype d'ISS.

L'un des fondements de la plate-forme Antelope est la préservation des ambiguïtés, permettant de retarder le choix définitif d'un sens jusqu'au moment où le système dispose de la meilleure connaissance possible du contexte. En effet, un enchaînement séquentiel de traitements accumule progressivement des informations sur le document analysé. Néanmoins, un composant d'analyse peut ne pas disposer individuellement de toutes les informations utiles pour prendre certaines

décisions et lever l'ambiguïté. Notre approche consiste à préserver les ambiguïtés en mémorisant les options possibles lors de chaque traitement ; on peut ainsi retarder le choix définitif, qui sera effectué lors d'une étape ultérieure, capable de s'assurer de la cohérence globale des contraintes locales en résolvant les éventuelles contradictions. Nous exposerons dans le chapitre B l'approche que nous proposons pour les gérer (localement puis globalement), en détaillant les traitements que nous avons expérimentés pour lever les ambiguïtés lexicales et syntaxiques.

Nous montrerons ensuite dans le chapitre C (page 163) comment extraire une ISS à partir (a) d'un analyseur en dépendance quelconque et interchangeable, (b) du lexique sémantique et (c) d'une base d'exemples associés aux représentations sémantiques que l'on souhaite calculer. Notre ISS est alors une grammaire de correspondance polarisée. Nous montrons en particulier comment obtenir un système modulaire, avec une *approche paresseuse*, en calculant certaines règles par « soustraction » de règles moins modulaires. Sans les rejeter, nous nous démarquons des approches statistiques qui prennent en compte la fréquence des constructions qu'elles extraient ; nous considérons au contraire que l'on peut extraire une règle à partir d'une seule occurrence (Cf. la méthode classique d'extraction des morphèmes à partir d'une paire minimale)[186], ce qui n'empêche pas ensuite de pondérer ces règles en fonction de leur fréquence d'utilisation dans l'analyse d'un corpus donné.

B. Gestion des ambiguïtés dans la plate-forme

L'ambiguïté est un objet polymorphe, omniprésent dans les langues naturelles et qui se manifeste à travers différents types de phénomènes. Nous recensons ici différents cas que nous avons rencontrés lors de l'implémentation des différents composants de traitement (sans prétendre à l'exhaustivité), en précisant dans chaque cas comment il est géré (ou non) par la plate-forme.

Nous commençons par présenter le mécanisme général de la plate-forme permettant de préserver les ambiguïtés en retardant leur levée. Nous précisons ensuite quels sont les phénomènes pris en compte par ce mécanisme. Nous finissons par ceux auxquels il ne s'applique pas, soit parce que l'ambiguïté peut être levée immédiatement dans la plupart des cas, soit au contraire parce que le phénomène est trop subtil pour être traité par la plate-forme.

1. Mécanisme permettant de préserver les ambiguïtés

Antelope permet de préserver, aussi longtemps que possible, l'ambiguïté des différentes unités linguistiques. Un mécanisme technique générique est utilisé avec un processus qui se déroule en trois phases.

Au début, un élément potentiellement polysémique est créé avec ses différents « sens », « analyses » ou « interprétations ». À ce stade, tout élément porteur d'ambiguïté a une liste de « candidats » possibles possédant chacun un score initial.

[186] Une paire comme *(nous) chantons* = CHANTER $_{\text{ind, prés, 1, pl}}$ vs *(nous) chantions* = CHANTER $_{\text{ind, imp, 1, pl}}$ suffit à faire l'hypothèse que l'imparfait est exprimé par *-i-* et le présent par un morphème zéro.

Ce score évolue ensuite, dans le cadre d'un vote effectué par les composants de traitements, qui appliquent successivement différentes heuristiques[187]. Précisons que la plate-forme est livrée en standard avec plusieurs heuristiques. L'utilisateur peut les étendre ou coder ses propres heuristiques ; il peut aussi choisir celles qui seront utilisées pour un traitement donné en paramétrant leur importance relative et leur ordre d'exécution.

En fin de processus, les candidats ayant le meilleur score sont retenus. Cette approche permet donc de retarder un choix définitif autant que possible, de façon à l'effectuer avec un maximum d'indices.

2. Cas d'ambiguïtés dont la levée est retardée

Un tel mécanisme permet de gérer les ambiguïtés pouvant apparaitre lors des opérations d'analyse syntaxique, de calcul de la forme de base d'un mot, de désambiguïsation lexicale, de reconnaissance d'entités nommées, de calcul des anaphores et enfin d'étiquetage du rôle sémantique des actants d'un prédicat.

a) Etiquettes morphosyntaxiques

L'étiquetage morphosyntaxique d'une phrase peut produire plusieurs sorties (de même que l'analyse syntaxique en dépendances). Par exemple, un sens de « *le boucher sale la tranche* » a pour paraphrase « *le boucher, qui est sale, tranche (la viande)* », un autre sens possible ayant le verbe « *saler* » pour tête. Quand un analyseur syntaxique de surface est en mesure de produire des sorties multiples pour une phrase, la plate-forme les associe à la phrase avec un éventuel score s'il est disponible.

b) Formes de base d'un mot

La (ou les) forme(s) de base d'un mot et la liste des sens possibles pour chaque forme de base sont initialisées à partir des données du lexique sémantique et de routines morphologiques. Dans "*I found the city*", le verbe peut être FOUND au présent (« *je fonde la ville* ») ou FIND au passé (« *j'ai trouvé la ville* »). La plate-forme associe à chaque mot les différentes formes de base possibles.

Il existe aussi des cas où une analyse globale permet de trouver la seule lemmatisation possible d'une forme, mais où une analyse locale (comme font généralement les lemmatiseurs) ne le permet pas. Dans la paire *Le diplomate reconduit à la frontière un espion russe* vs. *Le diplomate reconduit à la frontière est un espion russe*, *reconduit* est alternativement une forme verbale finie et un participe passé du verbe RECONDUIRE. Il est donc essentiel de pouvoir maintenir les deux lemmatisations jusqu'à l'analyse syntaxique si l'on veut espérer trouver la bonne analyse dans les deux cas.

c) Ambiguïté lexicale

Un vocable peut avoir plusieurs sens. La section 5 ci-dessous détaille différentes heuristiques de désambiguïsation lexicale que nous avons expérimentées. Le sens le plus pertinent est cherché parmi ceux énumérés dans un lexique de référence. Or ce dernier n'est jamais exhaustif ; comment faire alors pour détecter de nouveaux sens de mots ? Le fait de connaître des règles de polysémie régulière et d'appliquer le mécanisme de coercition décrit page 78 (section IV.C.4.d) permet dans certains cas d'inférer dynamiquement un nouveau sens en fonction du contexte.

---

[187] (Carré *et al.*, 1991) définit l'heuristique comme « une règle qu'on a intérêt à utiliser en général, parce qu'on sait qu'elle conduit souvent à la solution, bien qu'on n'ait aucune certitude sur sa validité dans tous les cas ».

d) Classes d'entités nommées

Des ambiguïtés peuvent aussi apparaître lorsqu'un composant de reconnaissance d'entités nommées sait associer une liste de classes possibles à une annotation. Dans la phrase « Thierry Mugler annonce le lancement de sa nouvelle gamme de parfums », même un humain peut légitimement hésiter : est-ce que Thierry Mugler est une référence à la société ou à son créateur[188] ?

e) Ambiguïté syntaxique

L'ambiguïté syntaxique est une propriété des phrases qui peuvent raisonnablement être interprétées de différentes façons. L'ambiguïté peut provenir d'un mot ayant deux parties du discours ou des homonymes. L'ambiguïté syntaxique se distingue de l'ambiguïté lexicale car elle provient non des différents sens qu'un mot (pris isolément) peut avoir, mais des différentes relations possibles entre les mots dans la structure d'une phrase. Ainsi, la phrase « il regarde manger la biche » peut signifier soit que la biche mange, soit que quelqu'un mange la biche.

Les analyseurs syntaxiques intégrés à Antelope associent à une phrase ses différents arbres syntaxiques, chacun d'eux étant pondéré par un coût ou une probabilité initiale. Pour éviter une explosion combinatoire, on peut fixer un nombre maximal d'arbres.

Des ambiguïtés syntaxiques artificielles (i.e. sur lesquelles un humain n'aurait *a priori* pas d'hésitation) apparaissent en cas de rattachements prépositionnels multiples, par exemple dans la configuration syntaxique V NP PP où un verbe, un syntagme nominal et un syntagme prépositionnel se suivent : la phrase « elle a vu l'homme avec des jumelles » peut être interprétée comme « elle a vu l'homme en utilisant des jumelles » ou « elle a vu un homme qui avait des jumelles ». Nous proposons en section 6 ci-dessous une heuristique pour résoudre les ambiguïtés de ce type.

f) Antécédents d'une anaphore

Le composant de résolution d'anaphores d'Antelope gère les ambiguïtés de la façon suivante : chaque anaphore, pronom (non pléonastique) ou groupe nominal déterminé, a une liste d'antécédents possibles (avec un accord en genre et nombre si c'est pertinent dans la langue considérée). A chaque antécédent est associé un score calculé par plusieurs heuristiques ; une autre information qui peut être utilisée pour choisir parmi ces antécédents est le nombre de mots séparant l'anaphore et l'antécédent.

L'extraction des chaînes de coréférences se fait ensuite par calcul des composantes connexes du graphe des anaphores ; une difficulté non gérée actuellement consiste à vérifier la cohérence globale de chaque chaîne de coréférence, c'est-à-dire de s'assurer qu'on n'introduit pas de contradiction.

g) Cadre de sous-catégorisation

L'étiquetage des rôles thématiques (Cf. section V.C.3, page 110) peut aussi déboucher sur l'identification de différents cadres de sous-catégorisation. Par exemple, dans « Brutus killed Caesar », l'utilisation d'une ressource comme VerbNet indiquera que Brutus peut être *Agent* ou *Instrument* de l'événement. Une phrase comme « il peint la nuit » est plus complexe ; on peut l'interpréter comme « il peint pendant la nuit » ou bien comprendre que la nuit est le thème de la peinture.

---

[188] On est confronté ici à un cas de métonymie régulière (SOCIETE créée par PERSONNE).

3. Cas d'ambiguïtés levées immédiatement

a) Langue du document

Actuellement, les applications écrites avec Antelope font l'hypothèse qu'un document est rédigé dans une seule langue. Un composant de détection de langue est appliqué au niveau du document, soit dans sa globalité, soit en se restreignant aux premières phrases (pour des raisons de performance).

La plate-forme est néanmoins prévue pour gérer le multilinguisme et l'information de langue peut éventuellement être raffinée au niveau du mot. Dans le futur, cela permettra par exemple de gérer des citations dans une langue autre que la langue principale du document ou des textes français contenant ponctuellement des termes techniques anglais. Plusieurs indices peuvent être utilisés pour détecter un document multilingue[189] : l'absence dans le lexique d'un mot dans la langue principale, mais sa présence dans une autre langue ; des marques typographiques, comme des italiques ou des guillemets, constituent aussi un bon indice.

b) Segmentation d'un document en phrases

(Grefenstette, 1994) rappelle que de nombreuses ambiguïtés existent lors de la segmentation d'un document en phrases. Les phrases se terminent par une ponctuation ; mais si le point d'exclamation ou le point d'interrogation sont généralement non-ambigus, il n'en va pas de même en ce qui concerne le point, qui ne marque pas forcément la fin une phrase. Par exemple, il peut apparaître dans un acronyme (« *S.A.R.L.* »), dans une abréviation ou dans un nom propre (« *John F. Kennedy* »).

Un système de règles est codé dans Antelope pour prendre en compte les cas les plus fréquents. Nous estimons que la segmentation s'effectue correctement dans 99 % des cas sur les corpus que nous avons eu à traiter dans des contextes industriels. Ce résultat nous semblant « suffisant », nous n'avons pas cherché, pour l'instant, à gérer d'ambiguïté de ce type.

c) Reconnaissance de mots composés

Une suite de mots peut représenter une unité lexicale, ou non, en fonction du contexte. Par exemple, « *je couvre la pomme de terre* » peut être compris comme « *je couvre de terre la pomme* » ou « *je couvre la pomme_de_terre* ». La plate-forme contient un composant de reconnaissance d'expressions multi-mots, qui vérifie si une suite donnée de mots existe dans le lexique sémantique et propose alors de les regrouper en une seule unité lexicale. Ce composant délègue la décision effective de regroupement des mots à l'application appelante, qui dispose généralement d'un contexte d'appréciation plus large que le composant. La plate-forme ne propose donc ici qu'un choix local et ne gère pas l'ambiguïté, contrairement à SxPipe (Sagot, Boullier, 2008) qui gère des entrées et sorties ambiguës (sous forme de graphes orientés acycliques).

4. Cas d'ambiguïtés non traitées

Finissons par donner quelques exemples de phénomènes qui nous semblent trop subtils pour être traités aujourd'hui par une application informatique. Ils nécessitent pour être résolus une prise en compte d'un contexte large et une véritable construction de l'état du monde :

[189] Un point non géré dans la plate-forme concerne les mots ayant des formes similaires dans des langues différentes, comme « pain » (baguette en français, douleur en anglais).

- « Le chat saute sur la table » : était-il dessus au début ? C'est un problème classique, bien connu notamment des traducteurs du français vers l'allemand.
- « Pierre a encore cassé sa montre » : parle-t-on d'une seule montre cassée à plusieurs reprises ou de plusieurs montres ?
- « Jean et Pierre ont écrit 6 livres » : ont-ils écrits 6 livres chacun ou 6 livres à eux deux ou 6 livres ensemble ? On a ici une ambiguïté de portée de quantifieurs et de composition des groupes nominaux quantifiés (Kahane, 2011).

5. Ambiguïté lexicale

(Ide, Véronis, 1998) rappelle que la désambiguïsation lexicale est une tâche complexe. Utiliser WordNet comme référence de notre lexique sémantique ne facilite pas cette tâche. La multiplication des nuances de sens a pour revers de la médaille de complexifier l'identification du « meilleur » sens ; (Véronis, 2001) souligne que la granularité de WordNet est souvent trop fine pour que même des humains s'accordent sur la bonne étiquette à donner à un mot. D'autre part, l'exploration d'un graphe riche et dense de grande taille doit être soumise à des conditions d'arrêt pour éviter une explosion combinatoire lors des recherches.

On peut distinguer plusieurs typologies d'algorithmes, selon qu'ils privilégient la précision ou le rappel. Les mesures de précision et de rappel qui apparaissent dans la présente section ont été effectuées sur le corpus anglais SemCor[190] qui a servi à tester nos implémentations.

a) Heuristiques privilégiant la précision

Certaines heuristiques sont spécialisées dans la reconnaissance d'un phénomène particulier ou peu fréquent. Elles sont alors très précises, mais leur rappel est faible. On peut classer dans cette catégorie :

- L'heuristique qui reconnaît le sens de « *Paris* » dans des constructions comme « *Paris, Texas* » ou « *Paris, France* » ; simple à implémenter, et s'appliquant dès l'analyse syntaxique de surface, elle offre une précision de 80,0 %.
- Le simple test de la capitalisation de l'initiale (précision = 88,5 %).
- Les restrictions de sens appliquées lors de l'étiquetage des rôles thématiques par application des cadres de sous-catégorisation (précision = 42,7 %).

b) Heuristiques privilégiant le rappel

D'autres heuristiques sont au contraire de portée très générale. Dans cette famille, l'algorithme consistant à prendre le premier sens dans WordNet d'un mot anglais sert souvent de base de comparaison. Son implémentation est particulièrement simple ; il propose toujours un résultat pour un nom commun (sauf quand le nom est inconnu du lexique), ce qui lui donne un rappel proche de 100 % ; nous avons évalué sa précision sur le corpus SemCor à 71,3 %.

C'est là que le bât blesse : nous n'avons pas pour l'instant trouvé d'algorithme général, applicable avec un rappel élevé, qui donne une meilleure précision que cette *baseline*. Notons que cet algorithme n'est pas facilement transposable au français ; en effet, dans une ressource comme WOLF, les différentes lexies d'un même vocable ne sont pas ordonnées.

[190] (Cf. IV.B.1.k) Ce sous-ensemble du corpus Brown compte 676 546 mots dont 234 135 noms, verbes, adjectifs et adverbes qui ont été annotés manuellement en sens par rapport à WordNet 1.6.

c) Algorithme de Lesk

L'autre algorithme privilégiant le rappel implémenté dans Antelope est (Lesk, 1986) enrichi pour WordNet, décrit dans (Banerjee, Pedersen, 2003) ; il consiste à compter le nombre de mots communs entre les définitions d'un mot et les définitions des mots de son contexte (ici, une fenêtre de quatre mots pleins à gauche et à droite du mot cible). Le sens retenu correspond à la définition pour laquelle on compte le plus grand nombre de mots communs avec le contexte. WordNet permet de généraliser cette approche en suivant les relations de synonymie et d'hyperonymie.

La précision obtenue avec cet algorithme, mesurée sur le corpus SemCor, est de 45,6 %. On peut se demander si une heuristique avec une précision aussi faible est vraiment utile. Peut-être faudrait-il identifier s'il existe un sous-ensemble particulier du lexique sur lequel cette heuristique offre une précision plus élevée ?

d) Combinaison de ces heuristiques

Notre approche actuelle de la désambiguïsation lexicale consiste à faire voter simultanément plusieurs algorithmes, pondérés par une importance relative. Ces différentes heuristiques peuvent être activées ou non, à la demande. Le résultat du vote est la combinaison linéaire des valeurs calculées par chaque heuristique, pondérées par son poids. Quand toutes les heuristiques sont combinées, la précision globale est de l'ordre de 55 % ; ce chiffre est décevant quand on le compare à l'heuristique qui consiste à simplement choisir le premier sens de WordNet pour l'anglais.

Pour essayer d'améliorer ces résultats, nous avons en cours de conception un algorithme d'apprentissage des relations syntaxiques du corpus SemCor ; sur des tests préliminaires, il offre une précision de l'ordre de 60 %. Une autre approche serait de ne prendre en compte que les heuristiques qui ont une précision supérieure à celle de la *baseline* (71,3 %), quitte à accepter un rappel faible. Concevoir et implémenter plusieurs algorithmes ne s'appliquant que dans un cas précis nécessiterait un important investissement, et il en faudrait un grand nombre pour augmenter sensiblement le rappel. Soulignons que dans l'ensemble des heuristiques de désambiguïsation lexicale donnant une précision élevée, la reconnaissance d'entités nommées est probablement celle qui offre le meilleur rappel sur un corpus spécialisé.

6. Ambiguïté syntaxique

Nous présentons ici des idées dont l'objectif est de contribuer à lever les ambiguïtés syntaxiques en utilisant des ressources externes à l'analyseur. Elles sont au stade de l'expérimentation, nous ne les évoquerons donc que brièvement.

a) Utilisation d'un étiqueteur morphosyntaxique comme « oracle » d'un analyseur syntaxique

Nous avons mené une expérience informelle autour de l'analyseur syntaxique du français FRMG (de la Clergerie *et al.*, 2009). L'expérience consistait à utiliser un second étiqueteur morphosyntaxique, indépendant de celui mis en œuvre en interne par FRMG, de façon à tenir compte d'une seconde source d'information pour calculer la partie du discours de chaque mot. Cela a permis d'améliorer la F-mesure de cet analyseur syntaxique d'environ 1 %. Dans le cas d'espèce, nous avons travaillé directement sur l'arbre de dérivation de FRMG. Cette approche n'est pas facilement généralisable à d'autres analyseurs syntaxiques ; en effet, ils n'exposent habituellement pas cette structure de données, qui n'est utilisée que comme intermédiaire de calcul.

b) *Désambiguïsation de la configuration syntaxique V NP PP*

(1) Principe

Nous avons évoqué le fait qu'une ambiguïté syntaxique est possible en cas de rattachements prépositionnels multiples. Nous proposons ici une heuristique pour aider à désambiguïser une configuration syntaxique particulière. Elle n'a été évaluée que dans des cas particuliers ; nous nous garderons donc d'en tirer des conclusions définitives, mais les résultats semblent prometteurs.

L'idée est de procéder à un « Google fight »[191] permettant d'évaluer le nombre d'occurrences de deux constructions syntaxiques mutuellement exclusives. Illustrons cela dans le cas d'une configuration comme V NP PP, en prenant comme exemple la phrase « manger une pizza avec X » ; la nature sémantique de X détermine si le syntagme prépositionnel se rattache au verbe[192] ou bien au chunk nominal qui le précède immédiatement[193].

Dans notre exemple, l'heuristique proposée revient donc essentiellement à chercher si on dit « pizza avec X » ou plutôt « manger avec X », pour un X donné. Faire un grand nombre de comparaisons de ce type soulève trois problèmes pratiques : il faut tenir compte de plusieurs variantes de surface de l'expression cherchée, disposer d'un corpus de référence de grande taille (pour que l'espace de recherche soit représentatif) et éviter de faire exploser le temps de calcul. Ce dernier point rend quasi impossible l'utilisation d'un moteur de recherche sur Internet, car le délai de latence d'une requête http unitaire est de l'ordre de 50 ms. Il faut donc privilégier l'usage d'une ressource locale.

(2) Ressource utilisée

L'utilisation de la ressource *Web 1T 5-gram Corpus* de Google dans le cadre de nos tests a permis d'apporter une solution aux problèmes de performance et de réprésentativité du corpus. Distribuée depuis septembre 2006 par le *Linguistic Data Consortium*, cette ressource a été constituée par Google à partir d'un corpus Web de 1 000 milliards de mots, venant de pages en principe en anglais. Elle donne les fréquences de toutes les combinaisons allant de 2 jusqu'à 5 mots, apparaissant plus de 40 fois dans le corpus.

L'intérêt de cette volumineuse ressource (24 giga-octets sous forme compressée), une fois stockée en local, est de permettre de faire très rapidement des recherches, sans limitation de volume. Nous avons développé un système d'index qui effectue chaque recherche élémentaire en 1 ou 2 millisecondes.

Sa limite est évidemment de n'autoriser des recherches que sur des groupes de 5 mots au maximum, c'est-à-dire des expressions courtes correspondant à des fragments de phrases. Cela n'a pas soulevé de réel problème dans nos tests, comme nous allons le voir.

---

[191] Cette opération (« confrontation sur Google » en anglais) consiste à comparer les résultats de deux requêtes en utilisant le même moteur de recherche pour déterminer celle des deux qui en renvoie le plus grand nombre.

[192] Comme dans :

- (VP manger (NP une pizza) (PP avec un ami))
- (VP manger (NP une pizza) (PP avec une fourchette))
- (VP manger (NP une pizza) (PP avec du vin))

[193] Comme dans :

- (VP manger (NP (NP une pizza) (PP avec du pepperoni)))
- (VP manger (NP (NP une pizza) (PP avec du jambon)))
- (VP manger (NP (NP une pizza) (PP avec du fromage)))

(3) Test sur une phrase en anglais

La ressource utilisée n'existant au moment de nos tests que pour l'anglais, nous avons effectué notre évaluation dans cette langue. En utilisant les routines morphologiques de la plate-forme, nous générons plusieurs n-grammes correspondant à des variantes de surface de l'expression cherchée. Sur notre exemple, le tableau 21 contient le nombre d'occurrence des variantes de "pizza with X", et le tableau 22 celui des variantes de "eat with X", dans le cas particulier où X = friend.

Expression cherchée	Nombre d'occurrences	Temps de recherche (ms)
pizza with a friend	139	1
pizza with my friend	42	1
pizza with friends	428	1
pizza with my friends	87	1
pizza with some friends	94	1
pizza with their friends	50	2
pizza with your friends	179	1
Total	1 019	8

Tableau 21 : Résultats de la recherche du nombre d'occurrences de « pizza with X »

Expression cherchée	Nombre d'occurrences	Temps de recherche (ms)
eat with a friend	372	1
eat with my friend	136	2
ate with a friend	63	2
ate with my friend	44	1
eat with friends	1 994	1
eat with her friends	89	1
eat with his friends	109	1
eat with my friends	375	2
eat with our friends	90	1
eat with some friends	249	2
eat with their friends	182	1
eat with your friends	366	2
ate with friends	130	1
eats with friends	41	1
Total	4 240	19

Tableau 22 : Résultats de la recherche du nombre d'occurrences de « eat with X »

On détermine donc (en 27 millisecondes) que les variantes cherchées de "pizza with X" apparaissent 1 019 fois et celles de "eat with X" 4 240 fois, dans le cas où X = friend. Pour autant que les variantes générées soit suffisamment représentatives des différentes constructions langagières possibles[194], nous trouvons donc ici un indice intéressant sur le fait que, dans "I eat the pizza with a friend", le groupe prépositionnel doit se rattacher au verbe et non au groupe nominal qui le précède.

(4) Tests sur plusieurs phrases

Nous avons appliqué ce système sur différentes phrases commençant par "I eat the pizza with…". Le tableau 23 compare les nombres de résultats trouvés sur "pizza with X" et "eat with X" pour

[194] Le travail exposé ici est préliminaire, et la génération des variantes mériterait évidemment une réflexion plus poussée.

différentes valeurs de X ; le plus grand des deux pour une ligne donnée est en gras ; quand le premier nombre est plus grand que le second, l'heuristique choisit un rattachement du syntagme prépositionnel en tant que complément de nom ; dans l'autre cas, le groupe prépositionnel est un complément du verbe. Sur ces phrases, les résultats sont tous satisfaisants.

Valeurs de X (PP "with X")	Nombre d'occurrences de "pizza with X"	Nombre d'occurrences de "eat with X"	Syntagme rattaché au PP
friend	1 019	**4 240**	V
fork	180	**2 939**	V
wine	0	**73**	V
pepperoni	**1 758**	0	NP
ham	**583**	149	NP
cheese	**2 066**	158	NP

Tableau 23 : Résultats du rattachement prépositionnel sur différentes phrases

(5) Conclusion sur cette heuristique

Nous avons mené des tests avec le Stanford Parser, capable de produire une forêt d'arbres. Pour tous les exemples précédemment cités, nous avons constaté que les deux rattachements sont toujours proposés, mais l'un des deux est invariablement présenté comme étant le plus probable. On voit en figure 60 une illustration de ce point avec les sorties produites par le Stanford Parser sur la phrase "I eat the pizza with a friend" : le rattachement du PP au NP est systématiquement proposé comme étant le plus vraisemblable.

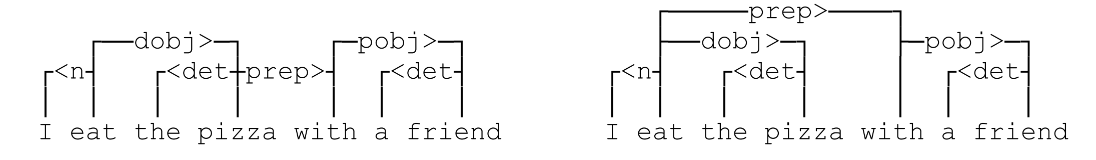


Figure 60 : Deux rattachements prépositionnels possibles sur une phrase de type V NP PP

L'heuristique que nous avons présentée ici semble donc intéressante pour lever l'ambiguïté de certaines configurations syntaxiques. Le temps de calcul n'est pas négligeable par rapport à celui de l'analyse syntaxique elle-même, mais rien n'empêche de mettre en place une optimisation, telle qu'un cache des résultats déjà calculés pour chaque « Google fight ».

Nous avons effectué nos tests en anglais. Le principe semble toutefois généralisable à d'autres langues, dont le français, du moment qu'on dispose d'une ressource de n-grammes constituée à partir d'un corpus de taille significative. Elle peut être calculée pour une langue donnée à partir de l'encyclopédie Wikipédia ou du projet Gutenberg par exemple ; notons que le projet Google Books Ngram propose le téléchargement direct de tels jeux de données pour l'anglais, le chinois, le français, l'allemand, l'hébreu, le russe et l'espagnol.

c) Reconnaissance d'expressions multi-mots

Disposer d'un lexique à large couverture permet aussi d'améliorer la précision d'un analyseur syntaxique capable de produire une forêt d'arbres, quand il traite une phrase qui contient des expressions multi-mots (Cf. section III.I.4, page 38). En effet, les constituants d'une telle expression sont regroupés à condition d'appartenir à un même sous-arbre. Comme le montre la figure 61, l'expression « *Battle of Gettysburg* » est reconnue en tant qu'unité lexicale dans l'arbre de gauche mais pas dans celui de droite car les différents mots n'y partagent pas de tête commune.

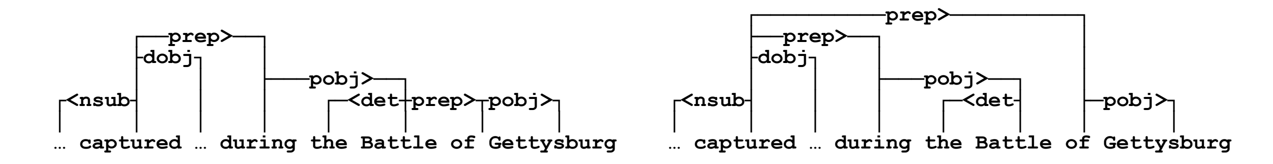


Figure 61 : L'identification d'expressions multi-mots permet de lever des ambiguïtés syntaxiques

Le composant qui identifie les expressions multi-mots contribue ainsi à la désambiguïsation syntaxique. En effet, il vote pour améliorer le score de l'arbre où il a reconnu une expression, et l'amélioration du score est proportionnelle au nombre de mots regroupés. L'hypothèse linguistique formulée ici est qu'il faut privilégier, toutes choses égales par ailleurs, les analyses syntaxiques permettant d'identifier les expressions multi-mots les plus longues.

7. Améliorer la prise en compte des ambiguïtés

Ce chapitre a présenté notre approche actuelle de la désambiguïsation : la plate-forme prend en compte plusieurs types d'ambiguïtés et implémente différentes heuristiques qui calculent un score pour chaque candidat possible ; ces différents choix sont mémorisés aussi longtemps que possible ; en fin de processus, les candidats ayant le meilleur score cumulé sont retenus. Cette démarche permet de repousser un choix définitif en fin d'analyse, de façon à disposer d'un maximum d'indices.

Nous souhaitons dans le futur améliorer ce système, qui reste limité à la juxtaposition d'un ensemble de choix locaux et manque donc d'une vision globale. Chaque composant d'analyse linguistique effectue sa tâche indépendamment des autres, en portant un jugement local. Par exemple, un module de désambiguïsation lexicale pourra attribuer à un nom (tel que « Paris » ou « Washington ») un sens de type lieu ; un module de résolution d'anaphore pourra considérer que ce même nom est l'antécédent d'un pronom désignant une personne ; les deux modules ne peuvent pas avoir raison simultanément… sauf si l'utilisation d'une figure de style comme une métonymie permet de personnifier le lieu dans le cas d'espèce.

Un enjeu futur important pour la plate-forme concerne donc sa capacité à fédérer ses différents composants de désambiguïsation sous la supervision d'un « chef d'orchestre », chargé de garantir la cohérence d'ensemble. Nous travaillons actuellement sur la formalisation des différents choix locaux sous forme de contraintes, compatibles ou non entre elles. L'ensemble des contraintes obtenues après l'analyse d'un document complet peut représenter un graphe de taille significative ; la recherche d'une solution optimale exacte satisfaisant toutes les contraintes risque alors de déboucher sur un temps de calcul prohibitif, du fait d'une explosion combinatoire ; nous explorons l'idée d'utiliser plutôt des algorithmes capables de calculer rapidement une solution approchée, comme ceux dits de « colonies de fourmis ». Une approche de ce type est proposée par (Schwab *et al.*, 2011) pour la désambiguïsation lexicale par propagation de mesures sémantiques locales.

C. Écrire et extraire une interface syntaxe-sémantique

Ce chapitre porte donc essentiellement sur des questions théoriques liées à l'écriture et à l'extraction d'une ISS, même si une implémentation est en cours. Il a fait l'objet d'une publication (Chaumartin, Kahane, 2010).

Le développement manuel d'une ISS peut être coûteux ; de plus, il est périlleux de construire une ISS qui s'appuie sur les sorties d'un analyseur syntaxique particulier qui peut rapidement devenir obsolète. Notre objectif est donc de pouvoir extraire une interface-sémantique automatiquement à partir de n'importe quel analyseur syntaxique. Notre idée est de partir d'une base de phrases d'exemples associées à leur représentation sémantique, de traiter ces exemples avec l'analyseur de notre choix, puis d'extraire une grammaire permettant de faire la correspondance entre ces arbres de dépendance et les graphes sémantiques qui leur sont associés.

Une des principales difficultés est d'arriver à obtenir la grammaire la plus modulaire possible et la plus couvrante sans multiplier les règles inutilement et sans avoir à fournir des quantités astronomiques d'exemples pour l'apprentissage. Nous avons évoqué au chapitre II.C (page 15) le type de représentation sémantique que nous considérons et le type de calculs qu'elle permet. Nous présenterons ici le formalisme que nous utilisons pour l'écriture d'une ISS (section 1), puis les principes généraux de l'extraction de règles grammaticales sans utiliser de connaissances lexicales (section 2). L'exploitation de ressources lexicales pour la production de nouvelles règles sera esquissée (section 3). Nous terminerons en montrant comment réaliser une interface lexique-grammaire par la « soustraction » de règles lexicales à nos règles grammaticales (section 4).

1. Écrire une ISS

Nous utilisons comme formalisme pour écrire notre ISS la Grammaire d'Unification Polarisée (GUP) (Kahane, 2004). Il permet d'écrire des grammaires de correspondances entre graphes et a déjà été proposé pour l'ISS (Kahane, Lareau, 2005). Il permet aussi, à l'image de TAG, de combiner des structures élémentaires afin d'obtenir une structure complète. Les structures que nous souhaitons obtenir sont des couples formés d'un arbre de dépendance syntaxique et d'un graphe sémantique ; nos structures élémentaires sont donc des fragments d'arbre syntaxique associés à des fragments de graphe sémantique et notamment des nœuds ou des dépendances syntaxiques associés à des nœuds ou des dépendances sémantiques.

La particularité de ce formalisme est un contrôle rigoureux de ce que chaque règle consomme, à l'aide de polarités associées aux objets manipulés par les règles. Le jeu de polarités le plus simple est constitué de deux polarités, que nous appelons noir (■) et blanc (□). Chaque objet de la structure reçoit une polarité. Sont considérés comme des objets les nœuds (identifiés avec l'élément lexical qu'ils portent), les dépendances et les éléments flexionnels ayant une contribution sémantique (temps verbal, nombre nominal, etc.). Les règles sont combinées par identification des objets dont les étiquettes peuvent s'unifier et les polarités se combiner.

Par exemple, la figure 62 présente un exemple d'ISS en GUP. En haut à gauche se trouve la phrase *Mary seems to sleep* avec l'analyse syntaxique en dépendance[195] qu'en propose le Stanford Parser. Nous ajoutons des polarités blanches sur les dépendances (☐) et les mots (☐) indiquant que ces objets doivent être consommés par des règles d'interface. Pour les verbes, une deuxième polarité blanche (○) indique que la flexion verbale doit aussi être consommée. En haut à droite se trouve le résultat attendu, c'est-à-dire un graphe sémantique associé à la phrase et polarisé en noir puisque produit par l'interface. Pour assurer cette correspondance, nous utilisons les trois règles qui figurent

[195] Les dépendances syntaxiques sont orientées vers la gauche (<nsubj) ou vers la droite (xcomp>).

en dessous[196]. Ce sont des règles lexicales associées aux lemmes SEEM, SLEEP et MARY. Comme on peut le voir la règle associée à SEEM consomme la totalité des dépendances syntaxiques, mais ne produit qu'une dépendance sémantique. La deuxième dépendance sémantique est produite par la règle de SLEEP. Mais pour que cette règle puisse s'appliquer il est nécessaire que la règle de SEEM restitue une dépendance syntaxique. Par souci de simplicité nous laissons de côté ici comme ensuite la question des étiquettes catégorielles sur les nœuds. Notons pour terminer que la règle de SEEM impose à son xcomp> d'être un verbe infinitif (Vinf) et consomme ainsi sa polarité flexionnelle (●).

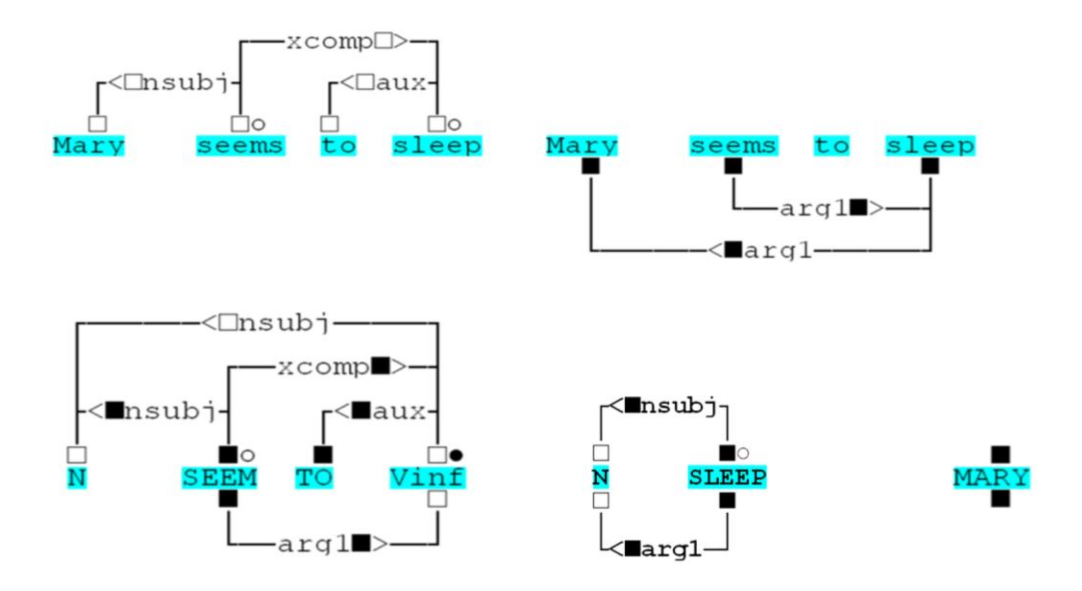


Figure 62 : Exemple d'interface syntaxe-sémantique en GUP

Notons qu'on peut avoir plusieurs types de polarités sur un même objet[197]. On peut par exemple aussi ajouter une polarité simplement pour contrôler que chaque nœud a au plus un gouverneur et vérifier ainsi que le graphe de dépendances syntaxiques est bien un arbre (Kahane, 2004).

2. Extraire une ISS

Une ISS prend en entrée les sorties d'un composant d'analyse syntaxique. Notre unique hypothèse est que nous partons d'analyses syntaxiques en dépendance ; nous ne souhaitons pas faire d'hypothèses sur la nature exacte des étiquettes des nœuds et des dépendances des arbres syntaxiques produits. Notre idée est donc d'écrire des « demi-règles » dont l'autre moitié sera calculée par l'analyseur de notre choix.

La première difficulté est d'identifier les traits de la structure syntaxique correspondant aux éléments sémantiques que nous souhaitons identifier. On peut bien sûr dresser une table de correspondance à la main, mais nous préférons extraire ces informations à partir d'exemples. Pour cela, nous adoptons les méthodes de l'analyse distributionnelle et le principe de commutation. Supposons qu'on veuille savoir quel est le trait indiquant le temps verbal dans l'analyse de *Mary **sleeps*** ; il suffit de comparer l'analyse de cette phrase avec celle de *Mary **slept*** et de considérer que ce qui a varié est l'expression

[196] Les analyses présentées ont été obtenues avec le Stanford Parser (c'est le cas pour la figure 62) ou le Link Grammar Parser.
[197] Voir (Kahane, Lareau, 2005) pour l'articulation de plusieurs modules à l'aide de polarités d'interface entre modules.

du temps verbal. Par exemple, en analysant ces deux phrases avec le Link Grammar, on voit que seule change l'étiquette catégorielle : /VBZ pour *sleeps* et /VBD pour *slept*. On obtient ainsi deux règles flexionnelles (Figure 63).



Figure 63 : Règle extraite concernant le temps verbal

De la même façon, pour identifier le trait porteur du lemme, il suffit de comparer ***Mary*** *left* avec ***Ann*** *left*. Nous ne souhaitons pas écrire une règle pour chaque lemme, mais avoir une règle très générale de copie de la valeur du lemme.

Voyons maintenant comment extraire la règle pour l'aspect progressif. Nous savons que l'aspect progressif est exprimé par BE + Ving et nous voulons récupérer au niveau sémantique un attribut [aspect=progressive] sur le verbe. Pour apprendre la règle, nous allons construire un exemple de phrase avec un progressif (en l'occurrence *Mary is sleeping*) en indiquant que pour *is* le lemme seul sera consommé (■○) et que pour *sleeping* la flexion seule sera consommée (□●) (voir figure 64). Par ailleurs nous devons indiquer quelles dépendances syntaxiques seront consommées. Or nous ne savons pas exactement ce que va faire l'analyseur : nous lui indiquons simplement une liste de nœuds (surlignés dans nos exemples) et nous considérons que tous les liens syntaxiques entre ces nœuds seront polarisés en noir dans la règle. Dans le cas du progressif illustré figure 64, nous ne savons pas si le sujet sera relié à l'auxiliaire ou au verbe lexical. Nos deux analyseurs de référence, le Stanford Parser et le Link Grammar, font d'ailleurs des choix différents. C'est pourquoi nous intégrons le sujet dans la règle et considérons donc une relation sémantique <arg1 correspondante. Nous verrons dans la section 4 comment « retirer » cette information.

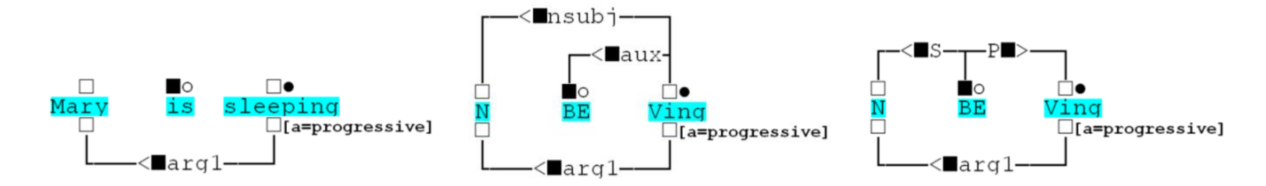


Figure 64 : Règle extraite concernant le progressif

3. Des règles lexicales pour l'ISS

L'utilisation d'un lexique électronique permet l'extraction automatique de règles. Par exemple, une ressource décrivant des cadres de sous-catégorisation (Cf. IV.B.3), telle que VerbNet pour l'anglais ou Dicovalence pour le français, peut être mise à profit. L'idée est alors d'analyser les exemples fournis pour en déduire les règles. Par exemple, le cadre give-13.1 de VerbNet est décrit de la façon suivante :

```
<DESCRIPTION descriptionNumber="0.2" primary="NP V NP PP.recipient"/>
<EXAMPLES><EXAMPLE>They lent a bicycle to me.</EXAMPLE></EXAMPLES>
<SYNTAX>
  <NP value="Agent" />
  <VERB />
  <NP value="Theme" />
  <PREP value="to" />
  <NP value="Recipient" />
</SYNTAX>
```

Cette description peut être utilisée pour créer automatiquement la règle lexicale de la figure 65, avec le processus décrit en section V.C.3 ; l'exemple de départ construit à partir de VerbNet est à gauche, la règle obtenue pour le Stanford Parser est à droite. Première étape : à partir de l'exemple donné par VerbNet et sa description dans VerbNet la demi-règle sémantique est construite. Deuxième étape : l'exemple est analysé par l'analyseur de notre choix (ici le Stanford Parser), ce qui nous fournit une règle lexicale pour l'interface avec les résultats de cet analyseur.

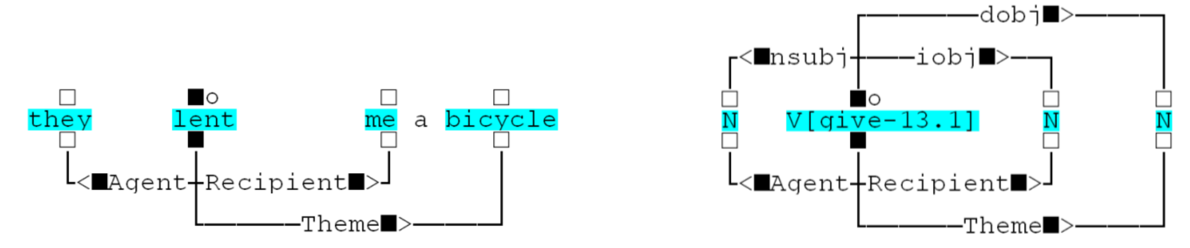


Figure 65 : Règle lexicale extraite partir du cadre give-13.1 de VerbNet

Cette règle est par ailleurs utile pour lever des ambiguïtés lexicales et syntaxiques : en effet, d'une part, la recherche de la règle la plus couvrante dans la forêt d'analyses syntaxiques produite pour une phrase donnée permet d'augmenter le score des analyses où la règle est applicable ; d'autre part, VerbNet précise les sens du verbe compatibles avec le cadre de sous-catégorisation et impose éventuellement des contraintes de sélection sur ses arguments. D'autres ressources électroniques, comme NomLex (qui décrit le cadre de sous-catégorisation des déverbatifs et la correspondance entre noms et verbes) ou un dictionnaire de locutions ou de collocations pourraient aussi être utilisées pour créer des règles.

4. Articulation lexique-grammaire et soustraction de règles

Considérons une phrase telle que *They were lending me a bicycle*. Nous pouvons y appliquer les règles grammaticales extraites en section 2 et notamment la règle du progressif. Mais cette règle consomme le lien sujet et nous ne pourrons pas y appliquer la règle lexicale du verbe LEND que nous avons créée à partir de VerbNet. La solution habituelle à ce problème est celle adoptée par exemple par les grammaires TAG consistant à produire à partir de la diathèse de base toutes les réalisations possibles (Candito, 1999). C'est par exemple la solution adoptée par (Bédaride, Gardent, 2009). Il en résulte un lexique-grammaire assez volumineux en raison de la croissance rapide du nombre de règles en fonction du nombre de phénomènes pris en compte (le lexique inclut en fait la grammaire).

Plutôt que d'additionner divers phénomènes au sein d'une même règle, nous proposons au contraire de soustraire aux règles grammaticales la partie lexicale pour permettre à la règle lexicale de se combiner avec les règles grammaticales. Prenons l'exemple du progressif, réalisé par une

construction avec un auxiliaire BE + Ving. Pour pouvoir appliquer sans problème une règle lexicale nous devons nous ramener au cas d'une forme simple. Pour ramener la construction A au cas d'une construction B, nous proposons simplement d'extraire comme précédemment des règles pour A et B puis de soustraire la règle de B à celle de A. La soustraction est contrôlée par les polarités selon le calcul suivant :

■ - ■ = suppression (autrement dit tout objet qui manipulé dans A et B est supprimé)

(□ -) ■ = □ (un objet uniquement manipulé dans B doit être absolument introduit dans la règle A–B pour être consommé ensuite par l'application de B)

■ (- □) = ■ (un objet manipulé uniquement dans A doit figurer dans A–B)

□ (- □) = □ (un objet qui est seulement dans le contexte des règles A et B reste dans le contexte)

Dans la figure 66, la première ligne contient les exemples A et B dont nous partons. La deuxième ligne montre les règles obtenues pour le Link Grammar : le lien <■S de B ne correspond pas au lien <■S de A et donne donc un lien <□S dans A–B. La troisième ligne montre les règles obtenues pour le Stanford Parser : ici, les liens <■nsubj de A et B se correspondent et s'annulent donc l'un l'autre.

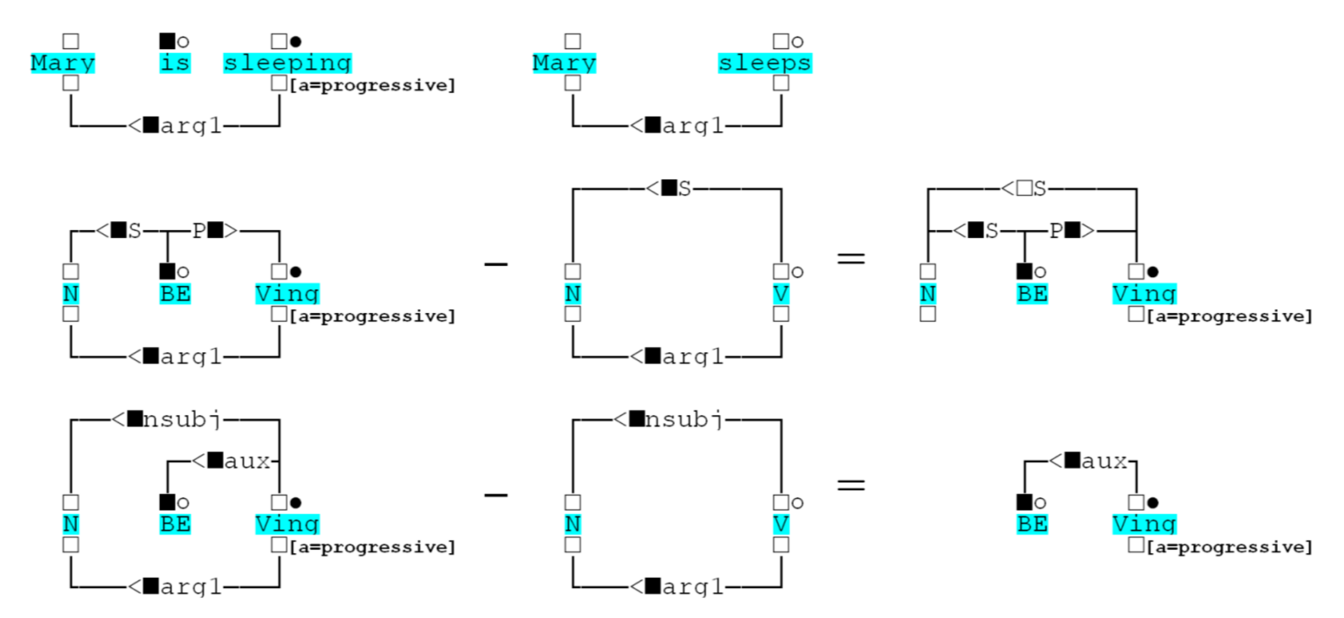


Figure 66 : Extraction de la règle pour le progressif

Nous présentons de la même façon les règles obtenues pour le passif (figure 67), une relative (figure 67) et une extraction non bornée (figure 67). Le principe est toujours le même : on soustrait à une règle A ce qui permettra d'appliquer ensuite la (ou les) règle(s) B. Dans le cas de la dépendance bornée, on soustrait ainsi des fragments de deux règles B différentes ; nous donnons cette règle pour le Link Grammar. On voit que celui-ci ne gère pas les dépendances non bornées, mais cela n'a pas d'importance puisqu'on les récupère lors de l'ISS. Comme on le voit, la règle qui fait ce calcul est en plus particulièrement simple : c'est une règle qui s'apparente à l'adjonction prédicative en TAG en permettant au verbe pont de venir s'intercaler dans la chaîne de dépendance qui lie l'antécédent au verbe recteur.

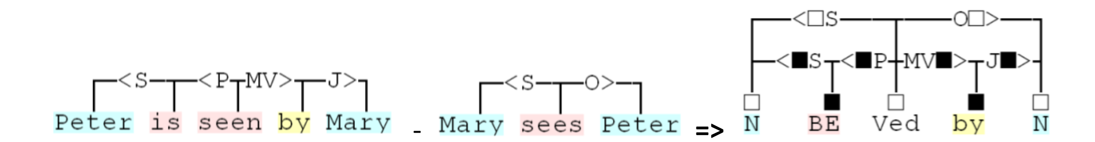


Figure 67 : Extraction de la règle pour le passif

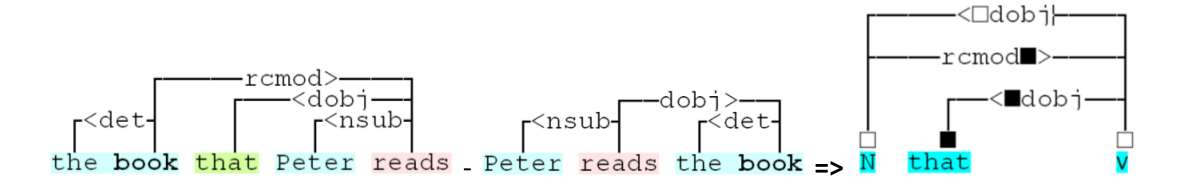


Figure 68 : Extraction de la règle pour les relatives

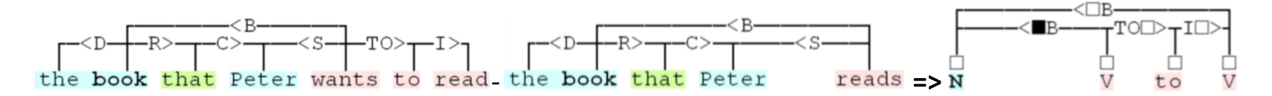


Figure 69 : Dépendances non bornées

5. Conclusion

Nous avons présenté une approche de la conception d'une ISS basée sur l'exemple. Notre approche sait prendre en compte des règles de différents niveaux, grammaticales (temps, passifs, relatives…) ou lexicales (sous-catégorisation). Notre choix est de procéder au calcul de règles élémentaires, par soustraction de règles, en évitant l'explosion combinatoire du nombre et de la complexité des règles que provoque une « précompilation » du lexique et de la grammaire. Nous pensons que cette approche est pragmatique dans la mesure où elle met en œuvre des ressources à large couverture dans leur état actuel. Les avantages escomptés de cette approche sont une grande modularité et une facilité de maintenance de l'ISS obtenue, l'indépendance vis-à-vis de tout analyseur syntaxique particulier et une facilité de prise en compte de nouvelles ressources.

Ce chapitre portait principalement sur l'extraction de règles pour l'ISS. Notons que notre grammaire est complètement réversible et peut servir aussi bien pour l'analyse que pour la génération de texte. La mise en œuvre d'une telle grammaire pose évidemment des difficultés qui ne sont pas abordées ici. Notons simplement que le formalisme a déjà fait l'objet d'une implémentation (Lison, 2006) ; nous en développons une nouvelle implémentation en testant différentes heuristiques afin d'éviter toute explosion combinatoire.

Partie VIII. Conclusion

A. Bilan

Voici donc résumé le travail de recherche et d'ingénierie effectué pendant quelques années. Notre objectif était de montrer qu'en créant une plate-forme de TAL, le développement d'applications sémantiques peut être simplifié et industrialisé. La plate-forme Antelope fédère aujourd'hui des composants d'analyse syntaxique et sémantique en les rendant interchangeables pour une tâche donnée ; elle intègre aussi un lexique sémantique multilingue à large couverture.

Disponible sans contrainte pour la recherche et l'enseignement, Antelope a été téléchargée (sur www.proxem.com) par plus de 2 500 internautes en décembre 2011. Compatible avec les principaux systèmes d'exploitation du marché (Windows et Linux[198]), cette plate-forme de traitement linguistique est encore en cours de développement mais d'ores et déjà utilisable. Nous estimons que la force de la plate-forme est d'être :

- Robuste : elle a fait ses preuves sur l'analyse de corpus totalisant plusieurs centaines de millions de mots.
- Simple à mettre en œuvre : elle est livrée avec un programme d'installation et une documentation complète (tutoriel, exemples de code, fichier d'aide). Un informaticien non linguiste peut intégrer des traitements syntaxiques et sémantiques complexes au sein d'un progiciel ou d'un système d'information d'entreprise.
- Extensible : un informaticien linguiste peut facilement implémenter des heuristiques spécifiques, et étendre le lexique sémantique avec ses propres données.
- Complète : les composants livrés en standard couvrent plusieurs des principales tâches classiques de TAL.
- Solide : elle a pour base théorique la TST.
- Riche : la plate-forme intègre un grand nombre de ressources libres, proposant ainsi en standard un lexique sémantique de plus de 400 000 entrées. Nous avons aussi développé des ressources propres à la plate-forme (par exemple, les relations de polysémie régulière). Toutes ces ressources sont interopérables, ce qui est le propre d'une plate-forme ; nous les avons converties dans un même format homogène. La plate-forme est régulièrement mise à jour et nous pouvons intégrer les mises à jour des ressources externes tout en conservant les corrections et modifications de format que nous avons effectuées dessus.

La conception de la plate-forme est allée de pair avec une réflexion méthodologique sur l'acquisition semi-supervisée de connaissances. Cette démarche permet de construire rapidement des extensions du lexique spécifiques à un domaine. Elle a été mise en œuvre sur plusieurs cas de figure concrets pour créer des applications capables d'analyser des textes et d'en extraire différents niveaux de représentation du sens, en fonction des objectifs recherchés.

[198] Antelope fonctionne en principe sous Mac OS avec MONO, mais le test reste à effectuer.

B. Perspectives

Nous souhaitons continuer à faire progresser Antelope. Nos axes prioritaires concernent le multilinguisme, la désambiguïsation et le modèle théorique de l'interface syntaxe-sémantique (ISS).

1. Multilinguisme

Initialement, la plate-forme traitait uniquement l'anglais. Nous avions amorcé la prise en compte du français dans Antelope en 2009, en intégrant l'analyseur syntaxique TagParser et la ressource lexicale WOLF (WordNet libre du français). A ce jour, l'anglais et le français sont pris en charge au niveau du lexique sémantique et des composants de traitement, avec un niveau de traitement sémantique comparable (modulo la couverture des ressources utilisées). L'équipe Proxem prévoit d'intégrer aussi à Antelope l'analyseur syntaxique du français FRMG (de la Clergerie *et al.*, 2009).

Une extension du lexique sémantique aux principales langues européennes (espagnol, portugais, italien et allemand) a démarré, notamment grâce aux versions en différentes langues de la Wikipédia et du Wiktionnaire. Elle devrait être finalisée d'ici courant 2013, et s'accompagner de l'intégration d'analyseurs syntaxiques de surface pour ces langues.

2. Désambiguïsation et ISS

Nos travaux en cours portent sur l'amélioration de l'ISS actuelle. Pour nous rapprocher progressivement de la représentation sémantique idéale que nous souhaitons obtenir, nous voulons améliorer les performances de l'ISS, à travers les actions suivantes :

- Exploiter les n-grammes de Google (Cf. VII.B.6.b) dans les tâches de désambiguïsation lexicale et syntaxique ainsi que pour la résolution d'anaphores.
- Introduire de nouvelles heuristiques de désambiguïsation utilisant un apprentissage.
- Prendre en compte simultanément plusieurs types d'ambiguïtés.
- Augmenter l'interaction et le partage de résultats entre composants de traitement, pour arriver à un meilleur système de coopération entre agents linguistiques.
- Approfondir le modèle théorique de règles de réécriture de graphes de l'ISS.
- Prévoir un paramétrage des règles de l'ISS qui permette d'intégrer de nouveaux analyseurs à moindre coût (comme présenté au chapitre VII.C).
- Continuer d'étendre le lexique sémantique en y intégrant de nouvelles ressources.
- Renforcer dans la plate-forme les passerelles entre TAL et intelligence artificielle ; une prochaine étape consiste à intégrer les ressources CYC et ConceptNet évoquées en IV.D.4.

3. Et après ?

Se rapprocher de notre objectif –rendre le texte calculable– force à cheminer sur une route longue, sinueuse et parfois étroite. L'intuition n'y est pas toujours la meilleure boussole, et ne remplace jamais une expérience avec une mesure des résultats. Quand on commence à emprunter cette route, on découvre un troublant « effet d'horizon » avec un objectif qui donne parfois le sentiment de s'éloigner alors qu'on s'en rapproche. Mais, comme dit le proverbe gitan, *ce n'est pas la destination mais la route qui compte*.

Références

A. Bibliographie

***Remarque** : le thème principal de chaque référence bibliographique est précisé le cas échéant par une étiquette en fin de référence, quand il concerne spécifiquement la reconnaissance d'entités nommées [NER], le regroupement de documents [CLUSTERING], l'extraction d'information [IE], l'interface syntaxe-sémantique [ISS], l'apprentissage automatique [ML], le Web sémantique [SW] ou les ressources humaines [RH] ; enfin, l'étiquette [CIT] indique une publication mentionnant Antelope.*

AMMARI A., DIMITROVA V., DESPOTAKIS D. (2011). Semantically Enriched Machine Learning Approach to Filter YouTube Comments for Socially Augmented User Models. Workshop on Augmented User Models, at the 19th Int. Conference on *User Modeling, Adaptation, and Personalization* (UMAP 2011), Girona, Spain. *[CIT]*

ANDREEVSKAIA A., BERGLER S. (2006). Mining WordNet for fuzzy sentiment: Sentiment tag extraction from WordNet glosses. Actes de *EACL-06, 11th Conference of the European Chapter of the Association for Computational Linguistics*, Trento, Italie.

APRESJAN J. (1974). Regular Polysemy. *Linguistics* 142, 5-32.

APRESJAN J. ET AL. (2003). ETAP-3 Linguistic Processor: a Full-Fledged NLP Implementation of the MTT. Actes de *MTT,* Paris, 279-288. [ISS]

AUER S., BIZER C., KOBILAROV G., LEHMANN J. (2007). DBpedia: A nucleus for a web of open data. In *The Semantic Web*, LNCS, Volume 4825/2007, pp. 722-735, Springer. [SW]

BAKER C., FILLMORE C., LOWE J. (1998). The Berkeley FrameNet project. Actes de *17th international conference on Computational linguistics*.

BANERJEE S., PEDERSEN T. (2003). Extended gloss overlaps as a measure of semantic relatedness, In *8th International Conference on Artificial Intelligence (IJCAI)*, Acapulco, Mexico.

BARQUE L. (2008). *Description et formalisation de la polysémie régulière du français*. Thèse de doctorat, Université Paris 7.

BARQUE L., CHAUMARTIN F.-R. (2008). La polysémie régulière dans WordNet. *Actes de TALN 2008*, Avignon.

BARQUE L., CHAUMARTIN F.-R. (2009). Regular Polysemy in WordNet. *Journal for Language Technology and Computational Linguistics (JLCL)* 24(2), pp. 5-18.

BEDARIDE P., GARDENT C. (2009). Semantic Normalisation: a Framework and an Experiment. Actes de *IWCS'09: 8th International Conference on Computational Semantics*, Tilburg, Netherland. [ISS]

BENTIVOGLI L., FORNER P., MAGNINI B., PIANTA E. (2004). Revising WordNet Domains Hierarchy: Semantics, Coverage, and Balancing. Actes de *COLING 2004 Workshop on Multilingual Linguistic Resources*, Genève, Suisse, pp. 101-108.

BERNERS-LEE T., HENDLER J., LASSILA O. (2001). The Semantic Web. In *Scientific American*, mai 2001. [SW]

BILHAUT F., WIDLÖCHER A. (2006). LinguaStream: An Integrated Environment for Computational Linguistics Experimentation. Actes de *11th Conference of the European Chapter of the Association of Computational Linguistics (Companion Volume)*, Trento, Italy.

BLUM A., MITCHELL T. (1998). Combining labeled and unlabeled data with co-training. Actes de *Workshop on Computational Learning Theory*, Morgan Kaufmann, p. 92-100. [ML]

BOHNET B., WANNER L. (2001). On using a parallel graph rewriting formalism in generation. Actes de *Workshop on Natural Language Generation, ACL 2001,* Toulouse. [ISS]

BOLLACKER K., EVANS C., PARITOSH P., STURGE T., TAYLOR J. (2008). Freebase: a collaboratively created graph database for structuring human knowledge. Actes de *ACM SIGMOD international conference on Management of data*, Vancouver, Canada. [SW]

BONFANTE G., GUILLAUME B., MOREY M., PERRIER G. (2010), Réécriture de graphes de dépendances pour l'interface syntaxe-sémantique. Actes de *TALN 2010*, Montréal, Canada. [ISS]

CAILLIAU F. (2010). *Des ressources aux traitements linguistiques : le rôle d'une architecture linguistique*. Thèse de doctorat, Université Paris 10.

CALZOLARI N., MC NAUGHT J., ZAMPOLLI A. (1996). *EAGLES Final Report: EAGLES Editors' Introduction.* EAG-EB-EI, Pisa.

CANDITO M.-H., KAHANE S. (1998). Can the derivation tree represent a semantic graph? An answer in the light of Meaning-Text Theory. Actes de *TAG+4*, Philadelphie, 21-24. [ISS]

CANDITO, M.-H. (1999). *Organisation modulaire et paramétrable de grammaires électroniques lexicalisées. Application au français et à l'italien*. Thèse de doctorat, Université Paris 7.

CARPENTER B. (2007). LingPipe for 99.99% Recall of Gene Mentions. Actes de *2nd BioCreative workshop*. Valencia, Spain.

CARRÉ R., DÉGREMONT J.F., GROSS M., PIERREL J.M., SABAH G. (1991). *Langage humain et machine*. Presses du CNRS.

CAZALS F., KARANDE C. (2008). A note on the problem of reporting maximal cliques. [CLUSTERING]

CHAMPEAU C. (2008). NLP in Java: A language detector.
Weblog: http://www.jroller.com/melix/entry/nlp_in_java_a_language (consulté en mai 2012).

CHAUMARTIN F.-R. (2006). Construction automatique d'une interface syntaxe-sémantique utilisant des ressources à large couverture en langue anglaise. Actes de *RECITAL*, Leuven, 729-735. [ISS]

CHAUMARTIN F.-R. (2007a). A knowledge-based system for headline sentiment tagging. Actes de *SemEval-2007 (ACL Workshop)*, Prague.

CHAUMARTIN F.-R. (2007b). Extraction de paraphrases désambiguïsées à partir d'un corpus d'articles encyclopédiques alignés automatiquement. Actes de *RECITAL*, Toulouse.

CHAUMARTIN F.-R., COUMONT E., MANCINELLI F., GRISEL O. (2009). Bridging Mono/.NET and Java in the SCRIBO Project: The Way to UIMA.NET. Actes de *RMLL*, Nantes.

CHAUMARTIN F.-R. (2008). ANTELOPE, une plate-forme industrielle de traitement linguistique. *Traitement Automatique des Langues* 49:2. [ISS]

CHAUMARTIN F.-R., KAHANE S. (2010). Une approche paresseuse de l'analyse sémantique ou comment construire une interface syntaxe-sémantique à partir d'exemples. Actes de *TALN*, Montréal. [ISS]

CHAUMARTIN F.-R. (2011). Proxem Ubiq : une solution d'e-réputation par analyse de feedbacks clients. Actes de *TALN*, Montpellier (session démonstrations industrielles).

CHAUMARTIN F.-R. (2012). Solution Proxem d'analyse sémantique verticale : adaptation au domaine des Ressources Humaines. Actes de *JEP-TALN-RECITAL*, Grenoble (session démonstrations industrielles). [RH]

CHENG D., KANNAN R., VEMPALA S., WANG G. (2006). A divide-and-merge methodology for clustering. *ACM Trans. Database System*, 31(4):1499–1525. [CLUSTERING]

CHOMSKY N., LASNIK H. (1993). Principles and Parameters Theory. In *Syntax: An International Handbook of Contemporary Research*. Berlin, de Gruyter.

CHUNG F. R. K. (1997). Spectral graph theory. *CBMS Regional Conference Series in Mathematics*, 92. [CLUSTERING]

CLÉMENT L., SAGOT B., LANG B. (2004). Morphology based automatic acquisition of large-coverage lexica. Actes de *LREC*, Lisbonne, Portugal (pp. 1841-1844).

COCH J. (1998). Interactive generation and knowledge administration in MultiMeteo. Actes de *9th Int. Workshop on Natural Language Generation (INLG'98)*, Niagara-on-the-Lake.

COPESTAKE A. (2009). Slacker semantics: Why superficiality, dependency and avoidance of commitment can be the right way to go. Actes d'*EACL 2009*, (pp. 1-9). Athens.

COPESTAKE A., BRISCOE T. (1995). Semi-productive polysemy and Sense Extension. *Journal of Semantics* 1, 15-67.

CORNUÉJOLS A., MICLET L., KODRATOFF Y. (2002). *Apprentissage Artificiel : Concepts et algorithmes*, Eyrolles. [ML]

CUNNINGHAM H., WILKS Y., GAIZAUSKAS R. (1996). GATE - a General Architecture for Text Engineering. Actes de *16th Conference on Computational Linguistics*, Copenhagen.

DAILLE B. (1994). *Approche mixte pour l'extraction de terminologie : statistique lexicale et filtres linguistiques*. Thèse en informatique fondamentale, Université Paris 7.

DANLOS L. (2005). ILIMP : Outil pour repérer les occurrences du pronom personnel il. Actes de *TALN*, Toulouse.

DANLOS L., SAGOT B. (2007). Comparaison du Lexique-Grammaire des verbes pleins et de DICOVALENCE : vers une intégration dans le Le*fff*. Actes de *TALN*, Toulouse.

DE LA CLERGERIE E., SAGOT B., NICOLAS L., GUENOT M.L. (2009). FRMG: évolutions d'un analyseur syntaxique TAG du français. Actes de *IWPT*, Paris.

DE SAUSSURE F. (1916). *Cours de linguistique générale*.

DEJONG, G. (1982). An overview of the FRUMP system. In W.G. Lehnert & M.H. Ringle (Eds.), *Strategies for Natural Language Processing*. Hillsdale: Lawrence Erlbaum. [IE]

DESPOTAKIS D. (2011). Multi-perspective Context Modelling to Augment Adaptation in Simulated Learning Environments. Proceedings of *User Modeling, Adaption and Personalization*. Lecture Notes in Computer Science, Springer-Verlag, Volume 6787/2011, pp. 405-408. [CIT]

DIJKSTRA E. W. (1965). Solution of a problem in concurrent programming control. In *Communications of the ACM*, septembre 1965, volume 8, p. 569.

DING C., H. Q. DING C., HE X., ZHA H., GU M., SIMON H. (2001). A Min-max Cut Algorithm for Graph Partitioning and Data Clustering. *Actes de 2001 IEEE International Conference on Data Mining,* 107–114. [CLUSTERING]

DOUMIT S., MINAI A. (2011). Online News Media Bias Analysis using an LDA-NLP Approach. Proceedings of *International Conference on Computational Science (ICCS) 2011*, Singapour, pp. 251-265. [CIT]

DUCLAYE F. (2003). *Apprentissage automatique de relations d'équivalence sémantique à partir du Web*. Thèse de doctorat, Télécom ParisTech.

ESULI A., SEBASTIANI F. (2006). SentiWordNet: A Publicly Available Lexical Resource for Opinion Mining. *Actes de LREC 2006, fifth international conference on Language Resources and Evaluation*, pp. 417-422.

FASS D. (1988). Metonymy and Metaphor: What's the difference? Actes de *Coling-88*, 177-181.

FEIGENBAUM L., HERMAN I., HONGSERMEIER T., NEUMANN E., ET STEPHENS S. (2007). The Semantic Web in Action. *Scientific American* vol. 297, pp. 90-97. [SW]

FELDMAN R., AUMANN Y., LIBERZON Y., ANKORI K., SCHLER J., ROSENFELD B. (2001). A domain independent environment for creating information extraction modules. Actes de *ACM International Conference on Information and Knowledge Management (CIKM)*, pp. 586-588. [IE]

FELLBAUM C. (1998). *WordNet, An Electronic Lexical Database*. Cambridge : MIT Press.

FELLBAUM C. (2000). Autotroponymy. In Ravin Y., Leacock C. (eds.): *Polysemy*, pp. 52-67. Cambridge: Cambridge University Press.

FERREIRA D., DA SILVA A. R. (2008). Wiki Supported Collaborative Requirements Engineering. Proceedings of *Wikis4SE'08 Workshop*, Porto, Portugal. [CIT]

FERRUCCI D., LALLY A. (2004). UIMA: an architectural approach to unstructured information processing in the corporate research environment. *Natural Language Engineering*, Volume 10, Issue 3-4, pp. 327-348.

FILLMORE C. (1968). The case for case. In Bach and Harms (Ed.): *Universals in Linguistic Theory*. New York: Holt, Rinehart, and Winston, pp. 1-88.

FITRIANIE S., YANG C.-K., DATCU D., CHITU A.G., ROTHKRANTZ L.J.M. (2010). Context-Aware Multimodal Human-Computer Interaction. *Interactive Collaborative Information systems*, volume 281, pp. 237-272. Springer, Studies in Computational Intelligence, 2010. [CIT]

FORT K., GUILLAUME B. (2007). PrepLex : un lexique des prépositions du français pour l'analyse syntaxique. Actes de *TALN*, Toulouse.

FOWLER, M. (2004). Inversion of control containers and the dependency injection pattern. Article en ligne, http://www.martinfowler.com/articles/injection.html (consulté en mai 2012).

FRANCOPOULO G. (2008). TagParser: on the way to ISO-TC37 conformance. Actes de *International Conference on Global Interoperability for Language Resources*, Hong Kong.

FRANCOPOULO G., DECLERCK T., SORNLERTLANVANICH V., DE LA CLERGERIE E., MONACHINI M. *(*2008). *Data Category Registry: morpho-syntactic and syntactic profiles*. Workshop: use and usage of language resource-related standards. LREC, Marrakech.

GAMMA E., HELM R., JOHNSON R., VLISSIDES J. (1993). Design patterns: Abstraction and reuse of object-oriented design. In *European Conference on Object-Oriented Programming Proceedings*, volume 707 of Lecture Notes in Computer Science, Springer-Verlag.

GREFENSTETTE G. (1994). What is a word, what is a sentence? Problems of Tokenization. COMPLEX'94, pages 79-87.

GRISHMAN R., SUNDHEIM B. (1996). Message Understanding Conference 6: A brief history. Actes de *International Conference on Computational Linguistics*.

GROSS M. (1994). Constructing Lexicon-grammars. In *Computational Approaches to the Lexicon*, Atkins and Zampolli (eds.), Oxford Univ. Press, pp. 213-263.

GRUBER J. (1965). *Studies in lexical relations*. Ph. D. Dissertation, MIT.

HASTIE T., TIBSHIRANI R., FRIEDMAN J. (2001). Hierarchical clustering. *The Elements of Statistical Learning*, 272–280, 2001. [CLUSTERING]

HEYER L.J., KRUGLYAK S., YOOSEPH S. (1999). Exploring Expression Data: Identification and Analysis of Coexpressed Genes. *Genome Research*, 11(9): 1106–1115, 1999. [CLUSTERING]

HIROCHI U., MEIYING Z., DELLA SENTA T. (1999). *The UNL, A Gift for a Millenium*, UNU/IAS, Tokyo.

HOBBS J. (1978). *Resolving Pronoun References*. *Lingua*, 44 : 311-338.

HOBBS J. ET AL. (1996). FASTUS: a cascaded finite-state transducer for extracting information from natural-language text. In *Finite State Devices for Natural Language Processing*. Cambridge, MA: MIT Press. [IE]

IBRAHIM A., KATZ B., LIN J. (2003). Extracting Structural Paraphrases from Aligned Monolingual Corpora. Actes de *Second International Workshop on Paraphrasing*.

IDE N., VÉRONIS J. (1998). Introduction to the special issue on word sense disambiguation: the state of the art. *Computational Linguistics*, 24(1):1-40.

IORDANSKAJA L., KITTREDGE R., POLGUÈRE A. (1988). Implementing a Meaning-Text Model for Language Generation. Actes de *COLING 1988*. [ISS]

JACKENDOFF R. (1972). *Semantic interpretation in generative grammar*. Cambridge, MA: MIT Press.

JACKIEWICZ A., HUNSTON S., EL-BÈZE M. (2010). Opinions, sentiments et jugements d'évaluation (préface). *Traitement Automatique des Langues*, 51:3, pp. 7-17.

JOUSSE F., GILLERON R., TELLIER I., TOMMASI M. (2006). *Champs conditionnels aléatoires pour l'annotation d'arbres*. [NER, ML]

KAHANE S., MEL'CUK I. (1999). Synthèse de phrases à extraction en français contemporain (du réseau sémantique à l'arbre syntaxique). *Traitement Automatique des Langues*, 40:2, pp. 25-85.

KAHANE S. (2002). *Grammaire d'Unification Sens-Texte : Vers un modèle mathématique articulé de la langue naturelle*, Document de synthèse de l'Habilitation à diriger des recherches, Université Paris 7. [ISS]

KAHANE S. (2004). Grammaires d'unification polarisées. Actes de *TALN*, Fèz. [ISS]

KAHANE S. (2011). Une modélisation des dites alternances de portée des quantifieurs par des opérations de combinaison des groupes nominaux. Actes de *TALN*, Montpellier.

KAHANE S., LAREAU F. (2005). Meaning-Text Unification Grammar: modularity and polarization. Actes de *MTT 2005*, Moscou. [ISS]

KILGARRIFF A., GAZDAR G. (1995). Polysemous relations. In Palmer F.R. (ed.). *Grammar and Meaning: Essays in Honour of Sir John Lyons,* pp. 1-25. Cambridge University Press.

KIPPER-SCHULER K. (2003). *VerbNet: a broad coverage, comprehensive, verb lexicon*. Thèse, University of Pennsylvania.

KLEIBER G. (1994). *Anaphores et pronoms*. Duculot, Louvain-la-Neuve.

LAFFERTY J., MCCALLUM A., PEREIRA F. (2001). *Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data*. [NER, ML]

LANGACKER R. (1969). On pronominalization and the chain of command. In Reibel and Schane (eds.) *Modern studies in English*, 160-186.

LAPPIN S., LEASS H.J. (1994). An algorithm for pronominal anaphora resolution. *Computational Linguistics*, p. 535-561.

LAVOIE B., KITTREDGE R., KORELSKY T., RAMBOW O. (2000). A Framework for MT and Multilingual NLG Systems Based on Uniform Lexico-Structural Processing. Actes de *6th Conference on Applied Natural Language Processing (ANLP)*, Seattle.

LENAT D. (1995). CYC: A large-scale investment in knowledge infrastructure. In *Communications of the ACM*, 1995.

LESK M. (1986). Automatic sense disambiguation using machine readable dictionaries: How to tell a pine cone from an ice cream cone. Actes de *Fifth International Conference on Systems Documentation*, ACM SIGDOC.

LEVIN B. (1993). English Verb Classes and Alternation: A Preliminary Investigation. Chicago, IL: University of Chicago Press.

LIN D. (1998). An information-theoretic definition of similarity. Actes de *15th International Conf. on Machine Learning*, p. 296-304.

LIN D., PANTEL D. (2001). DIRT - Discovery of Inference Rules from Text. Actes de *ACM SIGKDD Conference on Knowledge Discovery and Data Mining*.

LISON P. (2006). *Implémentation d'une interface sémantique-syntaxe basée sur des grammaires d'unification polarisées*. Master's thesis, Université Catholique de Louvain, Louvain-la-Neuve, Belgium. [ISS]

LITKOWSKI K. (2002). Digraph Analysis of Dictionary Preposition Definitions. Actes de *SIGLEX/SENSEVAL Workshop on Word Sense Disambiguation: Recent Successes and Future Directions*, Philadelphia.

LIU H., SINGH P. (2004). ConceptNet: A Practical Commonsense Reasoning Toolkit. *BT Technology Journal* vol. 22, pp. 211-226.

LOTH R., BATTISTELLI D., CHAUMARTIN F.-R., DE MAZANCOURT H., MINEL J.-L., VINCKX A. (2010). Linguistic information extraction for job ads (SIRE project). Actes de *RIAO 2010, 9th international conference on Adaptivity, Personalization and Fusion of Heterogeneous Information*, Paris. *[RH]*

MACLEOD C., GRISHMAN R., MEYERS A., BARRETT L., REEVES R. (1998). Nomlex: A lexicon of nominalizations. Actes de *Euralex'98*.

MAGNINI B., CAVAGLIÀ G. (2000). Integrating Subject Field Codes into WordNet. Actes de *LREC-2000, Second International Conference on Language Resources and Evaluation*, Athènes, Grèce, pp. 1413-1418.

MANNING C., KLEIN D. (2002). Fast Exact Inference with a Factored Model for Natural Language Parsing. *Advances in Neural Information Processing Systems 15* (NIPS).

MARTIN R. (1972). Esquisse d'une analyse formelle de la polysémie. *Travaux de linguistique et de littérature* 10, 125-136.

McCallum A., Li W. (2003). Early Results for Named Entity Recognition with Conditional Random Fields, Feature Induction and Web-Enhanced Lexicons. Actes de *CoNLL*. [ner, ml]

Mel'cuk I. (1988a). *Dependency Syntax: Theory and Practice*, SUNY Press, Albany. [iss]

Mel'cuk I. (1988b). Paraphrase et lexique dans la théorie linguistique Sens-Texte : vingt ans après, *Revue internationale de lexicologie et lexicographie*, Vol. 52/53, pp. 5-50/5-53. [iss]

Messiant C., Gabor K., Poibeau T. (2010). Acquisition de connaissances lexicales à partir de corpus : la sous-catégorisation verbale en français. *Traitement Automatique des Langues*, 51:1, pp. 65 à 96.

Meyers A., Reeves R., Macleod C., Szekely R., Zielinska V., Young B., Grishman R. (2004). The NomBank Project: An Interim Report. Actes de *HLT-NAAC*.

Mihalcea R., Moldovan D. (2001). eXtended WordNet: Progress Report. Actes de *NAACL Workshop on WordNet and Other Lexical Resources*, Pittsburgh, PA.

Milicevic J. (2007). *La paraphrase - Modélisation de la paraphrase langagière*. Bern : Peter Lang.

Miller G. (1995). WordNet: A lexical database. In *Communications of the ACM*, novembre 1995, pp. 39-41.

Minsky M. (1974). A Framework for Representing Knowledge. MIT-AI Laboratory Memo 306. Réimprimé dans *The Psychology of Computer Vision*, P. Winston (Ed.), McGraw-Hill, 1975.

Mitkov R. (1998). Robust pronoun resolution with limited knowledge. *COLING*, Montréal.

Moens, M.-F. (2006). *Information Extraction: Algorithms and Prospects in a Retrieval Context* (The Information Retrieval Series 21). New York: Springer. [ie]

Moldovan D., Novischi A. (2002). Lexical Chains for Question Answering. Actes de *COLING*.

Nazarenko A. (2004). *Donner accès au contenu des documents textuels - Acquisition de connaissances et analyse de corpus spécialisés*. Habilitation à Diriger les Recherches, Université Paris Nord.

Nguyen T., Phung D., Adams B., Tran T., Venkatesh S. (2010). Classification and Pattern Discovery of Mood in Weblogs. *Advances in Knowledge Discovery and Data Mining*. Lecture Notes in Computer Science, Springer-Verlag, Volume 6119/2010, pp. 283-290. [cit]

Niles I., Pease A. (2003). Linking Lexicons and Ontologies: Mapping WordNet to the Suggested Upper Merged Ontology. Actes de *International Conference on Information and Knowledge Engineering (IKE'03)*, Las Vegas, Nevada.

Osiński S., Stefanowski J., Weiss D. (2004). Lingo: Search Results Clustering Algorithm Based on Singular Value Decomposition. Advances in Soft Computing, Intelligent Information Processing and Web Mining. Actes de *International IIS: IIPWM´04 Conference*, 359—368. [clustering]

Ostler N., Atkins B. (1991). Predictable Meaning Shift: Some Linguistic Properties of Lexical Implication Rules. In Pustejovsky J., Bergler S. (eds.), *Lexical Semantics and Knowledge Representation: First SIGLEX Workshop Proceedings.* Berlin : Springer-Verlag.

PEDERSEN T., PATWARDHAN S., MICHELIZZI J. (2004). WordNet::Similarity - Measuring the Relatedness of Concepts. Actes de *Nineteenth National Conference on Artificial Intelligence (AAAI-04)*, San Jose, CA.

PETERS W. (2006). In Search for More Knowledge: Regular Polysemy and Knowledge Acquisition. Actes de *GWC*.

POIBEAU T. (2003). *Extraction automatique d'information : Du texte brut au web sémantique*. Paris : Lavoisier. [IE, SW]

POLANYI L., ZAENEN A. (2006). Contextual Valence Shifters. In J. G. Shanahan, Y. Qu, and J. Wiebe (eds.), *Computing Attitude and Affect in Text: Theory and Application*. Springer Verlag.

PORTER M.F. (1980). An algorithm for suffix stripping, *Program*, 14(3) pp 130−137.

POTTIER B. (1992). *Sémantique générale*. Paris, PUF.

PUSTEJOVSKY J. (1995). *The Generative Lexicon*. Cambridge : MIT Press.

QUILLIAN M. R. (1968). Semantic memory. In M. Minsky, (Ed), *Semantic information processing*, pp. 216-260. Cambridge, MA : MIT Press.

RAMSHAW L., MARCUS M. (1995). *Text Chunking Using Transformation-Based Learning*. In Yarovsky D. and Church K. (eds.). Actes de Third Workshop on Very Large Corpora. Association for Computational Linguistics, Somerset, New Jersey, pp. 82–94.

RATNAPARKHI A. (1996). A maximum entropy part-of-speech tagger. Actes de Actes de *Empirical Methods in Natural Language Processing Conference*, Univ. of Pennsylvania.

RESNIK P. (1995). Using Information Content to evaluate semantic similarity in a taxonomy. Actes de *IJCAI-95*, 448–453.

ROUILLARD J., TARBY J.-C. (2011). How to communicate smartly with your house? *International Journal Ad Hoc and Ubiquitous Computing*, Volume 7, No. 3, pp. 155-162. [CIT]

ROSSET S., GROUIN C., ZWEIGENBAUM P. (2011) Entités nommées structurées : guide d'annotation Quaero. Notes et documents LIMSI n°2011-04. [NER]

RUIZ-CASADO M., ALFONSECA E., CASTELLS P. (2005). Automatic assignment of Wikipedia encyclopedic entries to WordNet synsets. Actes de *AWIC*, 380-386.

SAGOT B., BOULLIER P. (2008). SxPipe 2 : architecture pour le traitement pré-syntaxique de corpus bruts. *Traitement Automatique des Langues* 49:2, pp. 155-188.

SAGOT B., FISER D. (2008). Construction d'un WordNet libre du français à partir de ressources multilingues. Actes de *TALN*, Avignon.

SAINT-DIZIER P. (2005). PrepNet: a Framework for Describing Prepositions: preliminary investigation results. Actes de *IWCS05*, Tilburg.

SALMON-ALT S. (2002). Le projet Ananas : l'annotation anaphorique pour l'analyse de corpus sémantiques. Actes du *Workshop CRAA – TALN*, Nancy.

SARAWAGI S., COHEN W. (2004). Semi-markov conditional random fields for information extraction. In *NIPs*. [NER, ML]

SCHWAB D., GOULIAN J., GUILLAUME N. (2011). Désambiguïsation lexicale par propagation de mesures sémantiques locales par algorithmes à colonies de fourmis. Actes de *TALN*, Montpellier.

SEKINE S., SUDO K., NOBATA C. (2002). Extended named entity hierarchy. Actes de *LREC*, îles Canaries, Espagne. [NER]

SERASSET G., BOITET C. (2000). On UNL as the future "html of the linguistic content" & the reuse of existing NLP components in UNL-related applications with the example of a UNL-French deconverter. Actes de *COLING*.

SHI J., MALIK J. (1997). Normalized Cuts and Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 22 : 888–905. [CLUSTERING]

SHI L., MIHALCEA R. (2005). Putting Pieces Together: Combining FrameNet, VerbNet and WordNet for Robust Semantic Parsing. Actes de *CICLing*, Mexico.

SLEATOR D., TEMPERLEY D. (1991). Parsing English with a Link Grammar. Actes de *Third International Workshop on Parsing Technologies*.

SOTO A., FLORES HERNÁNDEZ J., DE LOS ÁNGELES BUENABAD ARIAS M., DIEZ G. (2009). Using Ontologies to generate Learning Objects automatically. Actes de *MICAI*, Guanajuato, Mexico. [CIT]

SOWA, J.F. (1976). Conceptual Graphs for a Database Interface, *IBM Journal of Research and Development*, 20(4), pp. 336-357.

STRAPPARAVA C., VALITUTTI A. (2004). WordNet-Affect: an Affective Extension of WordNet. Actes de *LREC*, Lisbonne, pp. 1083-1086.

STRAPPARAVA C., MIHALCEA R. (2007). SemEval-2007 Task 14: Affective Text. Actes de *SemEval-2007 (ACL Workshop)*, Prague.

SUCHANEK F., KASNECI G., WEIKUM G. (2007). YAGO - A Core of Semantic Knowledge. Actes de *16th international World Wide Web conference* (WWW 2007), Banff, Canada. [SW]

SUTTON C., MCCALLUM A. (2006). An Introduction to Conditional Random Fields for Relational Learning. *Introduction to Statistical Relational Learning*. [NER, ML]

SZOLOVITS P. (2003). Adding a Medical Lexicon to an English Parser. Actes de *AMIA 2003* Annual Symposium: 639-643.

TANGUY L., HATHOUT N. (2002). Webaffix : un outil d'acquisition morphologique dérivationnelle à partir du Web. Actes de *TALN*, Nancy.

TANNIER X. (2006). *Traitement automatique du langage naturel pour l'extraction et la recherche d'informations*. Rapport de recherche 2006-400-006. Saint-Etienne : ENSM. [IE]

TESNIERE L. (1959). *Éléments de syntaxe structurale*, Paris, Klincksieck.

TRUYEN T.T., PHUNG D. (2008). *A Practitioner Guide to Conditional Random Fields for Sequential Labelling*. Notes de cours. Curtin University of Technology, Australie. [ML]

TSUR O., DAVIDOV D., RAPPOPORT A. (2010). A Great Catchy Name: Semi-Supervised Recognition of Sarcastic Sentences in Product Reviews. Actes de *ICWSM*, Washington.

TSURUOKA Y., TSUJII J. (2005). Bidirectional Inference with the Easiest-First Strategy for Tagging Sequence Data. Actes de *HLT/EMNLP*, pp. 467-474.

VALITUTTI A., STRAPPARAVA C., STOCK O. (2004). Developing Affective Lexical Resources. In *Psychology Journal*, 2(1).

VAN DE CRUYS T. (2010). Mining for meaning: the extraction of lexico-semantic knowledge from text. Thèse de doctorat, University of Groningen.

VAN DEEMTER K., KIBBLE R. (2000). On Coreferring: Coreference annotation in MUC and related schemes. *Computational Linguistics* 26(4), pp. 615-623.

VAN DEN EYNDE K., MERTENS P. (2003), La valence : l'approche pronominale et son application au lexique verbal. *Journal of French Language Studies* 13, 63-104.

VAN WILLEGEN I., ROTHKRANTZ L., WIGGERS P. (2009). Lexical Affinity Measure between Words. Proceedings of the *12th International Conference on Text, Speech and Dialogue*. Lecture Notes in Computer Science, Springer-Verlag, Volume 5729/2009, pp. 234-241. [CIT]

VARGA E., FURLAN B., JAKUS G., MILUTINOVIĆ V. (2010). Document Filter Based on Extracted Concepts. *Transactions on Internet Research* 6:1, pp. 5-9. [CIT]

VEALE T. (2006). A typology of Lexical Analogy in WordNet. Actes de *Global WordNet Conference*, Jeju, Corée.

VÉRONIS J. (2001). Sense tagging: does it make sense? In *The Corpus Linguistics Conference*. Lancaster, UK.

VICTORRI B., FUCHS C. (1996). *La polysémie – Construction dynamique du sens*. Paris, Hermès.

WALLACH H.M. (2004). *Conditional Random Fields: An Introduction*. [NER, ML]

XU W., LIU X., GONG Y. (2003). Document clustering based on non-negative matrix factorization. Actes d'*ACM SIGIR conference on Research and development in information retrieval*, 267–273. [CLUSTERING]

ZIDOUNI A., GLOTIN H., QUAFAFOU M. (2009). Recherche d'Entités Nommées dans les Journaux Radiophoniques par Contextes Hiérarchique et Syntaxique. Actes de *CORIA 2009 - Conférence en Recherche d'Information et Applications*. [NER]

B. Ressources

ACE (Automatic Content Extraction) – http://www.itl.nist.gov/iad/mig//tests/ace/

BalkaNet – http://www.ceid.upatras.gr/Balkanet/

CiteSeer – http://citeseer.ist.psu.edu

Colt – http://acs.lbl.gov/~hoschek/colt/

ConceptNet – http://conceptnet5.media.mit.edu/

COSMO – http://micra.com/COSMO

CRF – http://crf.sourceforge.net (package Java)

DBpedia – http://dbpedia.org

Dicouebe (MEL'CUK, POLGUÈRE) – http://olst.ling.umontreal.ca/dicouebe/

Dicovalence (MERTENS, VAN DEN EYNDE) – http://bach.arts.kuleuven.be/dicovalence/

eXtended WordNet (MIHALCEA, MOLDOVAN) – http://xwn.hlt.utdallas.edu

FrameNet (BAKER, FILLMORE, LOWE) – http://framenet2.icsi.berkeley.edu

FreeBase – http://www.freebase.com

GATE (CUNNINGHAM *ET AL.*) – http://gate.ac.uk

Global WordNet – http://www.globalwordnet.org

Google Books Ngram – http://books.google.com/ngrams/datasets

GrGen – http://www.info.uni-karlsruhe.de/software/grgen/

IKVM.NET – http://www.ikvm.net

JLangDetect (CHAMPEAU) – http://code.google.com/p/jlangdetect

Le*fff* (CLEMENT *ET AL.*) – Lexique des Formes Fléchies du Français – http://atoll.inria.fr/~sagot/lefff.html

LingPipe (CARPENTER) – http://alias-i.com/lingpipe/

LinguaStream (BILHAUT, WIDLÖCHER) – http://www.linguastream.org

Link Grammar Parser (SLEATOR, TEMPERLEY, LAFFERTY) – http://bobo.link.cs.cmu.edu/link

MONO – http://www.mono-project.com

NomBank (MEYERS *ET AL.*) – http://nlp.cs.nyu.edu/meyers/NomBank.html

NomLex (MACLEOD *ET AL.*) – http://nlp.cs.nyu.edu/nomlex/index.html

OpenCalais – http://www.opencalais.com

OpenCyc (Lenat) – http://www.opencyc.org

OpenNLP – http://incubator.apache.org/opennlp/

PrepLex (Fort, Guillaume) – http://loriatal.loria.fr/Resources/PrepLex.txt

Protégé – http://protege.stanford.edu

ResearchCyc (Lenat) – http://research.cyc.com

Roget's Thesaurus (dans le cadre du projet Moby) – http://icon.shef.ac.uk/Moby

SemCor Corpus – http://www.cs.unt.edu/~rada/downloads.html

SentiWordNet (Esuli, Sebastiani) – http://sentiwordnet.isti.cnr.it

Stanford Parser (Manning, Klein) – http://nlp.stanford.edu/software/lex-parser.shtml

SUMO (Niles, Pease) – http://www.ontologyportal.org – http://ontology.teknowledge.com

The Preposition Project (Litkowski) – http://www.clres.com/prepositions.html

UIMA – http://incubator.apache.org/uima – www.research.ibm.com/UIMA

UNL – Universal Networking Language (Hiroci *et al.*) – http://www.undl.org

VerbAction (Hathout et al.) – http://redac.univ-tlse2.fr/lexicons/verbaction.html

VerbNet (Kipper, Schuler) – http://verbs.colorado.edu/~mpalmer/projects/verbnet.html

Google Web 1T 5-gram Corpus (Franz, Brants) –
http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp ?catalogId=LDC2006T13

Web sémantique (standards W3C) – http://www.w3.org/standards/semanticweb/

Wikipédia en anglais – http://en.wikipedia.org

Wikipédia en anglais simplifié (*Simple Wikipedia*) – http://simple.wikipedia.org

Wikipédia en français – http://fr.wikipedia.org

WordNet (Miller, Fellbaum) – http://wordnet.princeton.edu

WordNet Domains & WordNet-Affects (Magnini, Cavaglià) – http://wndomains.itc.it/download.html

WordNet : correspondance entre versions – http://www.cs.unt.edu/~rada/downloads.html#wordnet et http://www.lsi.upc.es/~nlp/tools/mapping.html

WordNet::Similarity (Pedersen *et al.*) – http://www.d.umn.edu/~tpederse/similarity.html

YAGO (Suchanek *et al.*) –http://www.mpi-inf.mpg.de/yago-naga/yago

Annexe I. Le Web sémantique

A. Introduction

Le Web sémantique n'étant pas encore bien connu des praticiens du TAL, nous avons jugé utile de faire ici une introduction à ces concepts et standards émergents. Les descriptions figurant dans cette annexe sont évidemment inspirées de celles présentées sur le site du W3C[199]. Nous essayons de les mettre en perspective, mais aussi de mettre en évidence les forces et faiblesses qui pourraient en accélérer ou en freiner l'adoption, au vu de l'expérience concrète que nous en avons.

Commençons par esquisser une présentation intuitive des évolutions d'usage entre le Web initial et le Web sémantique. Aux débuts du Web (qualifié *a posteriori* de Web 1.0), l'internaute venait lire de l'information ou effectuer un achat sur un site d'e-commerce, sans réelle interactivité : « *j'achète une pizza en ligne* ». Le Web 2.0 a marqué une évolution importante concernant aussi bien les technologies employées (application cliente riche) que les usages ; les internautes sont désormais capables d'interagir entre eux et avec le contenu des pages ; on est passé à « *je fais une pizza-party à la maison, et mes amis votent en ligne pour la date qui leur convient* ». Le Web sémantique est un nouveau saut technologique qui offre une meilleure connaissance de l'information en temps réel, permettant d'automatiser des scénarios complexes en dotant les applications de capacités de raisonnement : « *mon assistant personnel organise une pizza-party dimanche midi en tenant compte du fait que Pierre est allergique au gluten, Paul rentre de voyage samedi et Marie est végétarienne* ».

Le Web sémantique utilise les fondements techniques du Web classique et ne remet pas en cause ce dernier. Il en étend les fonctions primaires : publier et consulter des documents. En revanche, les documents traités par le Web sémantique contiennent non pas des textes en langage naturel, mais des informations formalisées pour être traitées automatiquement. L'objectif à long terme de cette évolution du Web actuel vise à étendre systématiquement les pages HTML (lisibles par un œil humain) afin qu'elles contiennent aussi des informations structurées (accessibles à la machine)[200]. Le corollaire sera la possibilité d'automatiser des tâches complexes, nécessitant aujourd'hui une action humaine ; par exemple, réserver un billet de train et un séjour à l'hôtel pour préparer un voyage pourra être très largement pris en charge par un agent personnel intelligent.

L'un des moyens d'arriver à cette évolution est la définition d'un certain nombre de standards émergents. Ils permettent de définir la façon dont des informations peuvent être représentées (RDF), leur structuration (RDFS), le contrôle de leur cohérence (OWL, RIF), ainsi que la façon de faire des requêtes complexes dessus (SPARQL). Ces standards émergents du Web sémantique ont de multiples intérêts en TAL, car ils permettent de représenter aussi bien des référentiels linguistiques (thésaurus, taxonomie ou ontologie) que des graphes complexes, comme ceux issus de résultats d'analyse (correspondant à l'extraction d'entités nommées ou de relations, par exemple).

[199] http://www.w3.org/standards/semanticweb/

[200] RDFa, une proposition du W3C, permet d'annoter des pages HTML existantes avec des données RDF.

(Feigenbaum *et al.*, 2007) effectue un premier bilan du chemin parcouru, six ans après l'article fondateur (Berners-Lee *et al.*, 2001). Cette publication présente des cas pratiques d'utilisation des technologies du Web sémantique et les solutions concrètes qu'elles apportent, en particulier dans les domaines des soins de santé et des sciences de la vie. Le choix de ces domaines pour illustrer les applications du Web sémantique n'est pas fortuit. En effet, ils ont depuis longtemps structuré les informations qu'ils manipulent sous forme de thésaurus ou d'ontologies (citons par exemple MeSH[201] et UMLS[202]), ce qui en facilite l'adaptation vers les technologies du Web sémantique.

B. Standards introduits par le Web sémantique

Un élément qui a fortement contribué au succès du Web actuel a été la standardisation des protocoles (HTTP, FTP, URI, SOAP…), langages (HTML, XML, XSL, CSS…) et formats (PNG, SVG…). Cette normalisation est due au *World Wide Web Consortium* (ou W3C). Cet organisme a défini un ensemble de nouveaux standards ouverts, formant la « pile » du Web sémantique, comme illustré en figure 70.

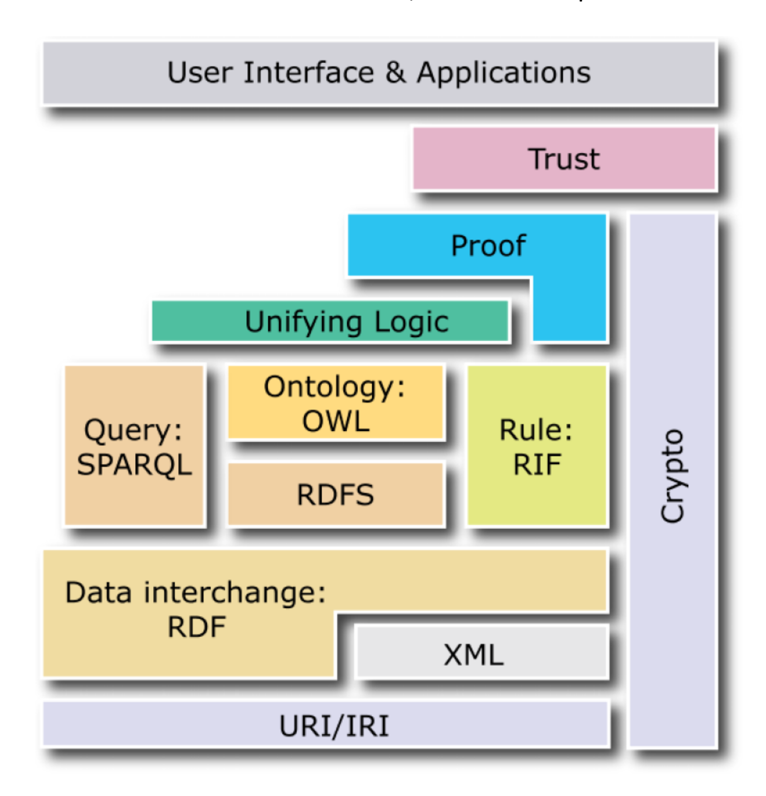


Figure 70 : La « pile » des standards du Web sémantique

Les plus importants de ces standards sont :

- RDF (*Resource Description Framework*) : un modèle conceptuel permettant de décrire toute information sous forme de triplets, produisant donc un graphe d'informations.

---

[201] MeSH (*Medical Subject Headings*) est le thesaurus de référence dans le domaine biomédical ; MeSH est utilisé par la base Medline/PubMed et sert d'outil d'indexation, de recherche et de classement.

[202] UMLS (*Unified Medical Language System*) est une compilation de nombreux vocabulaires contrôlés en sciences biomédicales. Il fournit une structure de correspondance entre ces vocabulaires et permet ainsi de les traduire en différents systèmes terminologiques. UMLS peut également être vu comme un thesaurus et une ontologie des concepts biomédicaux.

- RDFS (*RDF Schema*) : un langage autorisant la création de vocabulaires, par définition de classes et de propriétés, permettant de structurer les données en RDF.
- SKOK (*Simple Knowledge Organization System*) : un langage de définition de taxonomies et de thésaurus.
- OWL (*Web Ontology Language*) : un langage permettant de créer des ontologies, servant de support aux traitements logiques (inférence, classification).
- SPARQL : un langage de requêtes pour manipuler des informations à partir de graphes RDF.
- RIF (*Rule Interchange Format*) : un format d'échange de règles de gestion.

Nous avons estimé les applications d'OWL en TAL sont suffisamment importantes pour mériter de lui dédier le chapitre suivant. Ce sera notamment l'occasion de détailler les différents niveaux des logiques de description, leur pouvoir d'expression et le mécanisme de raisonnement.

1. RDF

RDF (*Resource Description Framework*) est un modèle de graphe destiné à décrire de façon formelle les ressources[203] Web et leurs métadonnées. L'objectif est de permettre le traitement automatique de telles descriptions. En annotant des documents non structurés et en servant d'interface vers le monde des données structurées, RDF permet l'interopérabilité entre applications échangeant de l'information sur le Web.

RDF/XML est l'une des syntaxes de ce langage permettant le stockage et les échanges sous forme XML. Notons que XML est un format de sérialisation possible pour les triplets RDF, mais pas le seul ; par exemple, les formats N3 et Turtle sont conçus pour être plus compacts et plus facilement lisibles par des humains.

a) Principe

Un document RDF est un ensemble de triplets. Chaque triplet RDF est une association {sujet prédicat objet} où[204] :

- Le sujet représente la ressource à décrire.
- Le prédicat représente un type de propriété applicable à cette ressource.
- L'objet est la valeur de la propriété, qui peut être soit une donnée, soit une autre ressource.

Le sujet et l'objet (dans le cas où l'objet est une ressource) peuvent être identifiés par une URI ou être des nœuds anonymes. Le prédicat est nécessairement identifié par une URI.

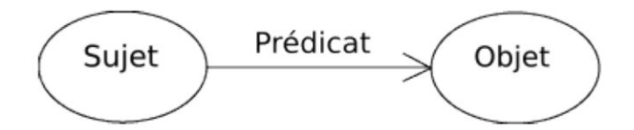
Un document RDF ainsi formé correspond à un graphe orienté étiqueté[205]. Chaque triplet correspond alors à un arc orienté dont le label est le prédicat, le nœud source est le sujet et le nœud cible est l'objet.

---

[203] La notion de ressource s'est progressivement étendue de son sens original de « document Web » (page HTML) à des sens plus généraux et plus abstraits. Dans les langages d'ontologie ou le langage SKOS, les ressources décrites sont des concepts comme des classes, des propriétés.

[204] On peut comprendre ces triplets de plusieurs façons équivalentes : {sujet prédicat objet}, {ressource propriété valeur}, {sujet verbe complément}…

[205] Notons que cette représentation de l'information sous forme de triplets n'est pas une nouveauté en tant que telle ; un atelier de génie logiciel tel qu'IEW (*Information Engineering Workbench*) utilisait une telle représentation dès les années 1990 pour proposer un métamodèle souple et extensible.



La sémantique d'un document RDF peut être exprimée en logique du premier ordre :

{sujet prédicat objet} ⇔ prédicat(objet, sujet)[206]

Partant du principe de base, « tout est ressource » (concepts abstraits, objets du monde réel, documents…), les éléments de description des identifiants, noms, attributs typés, relations sémantiques… sont tous exprimés selon le même schéma de triplets {sujet prédicat objet}. RDF définit donc un modèle de données abstrait indépendant de toute syntaxe ou mode de stockage.

b) *Exemple*

Par exemple, le triplet {Victor_Hugo auteur_du_livre Les_Misérables} se transpose ainsi en RDF :

- Sujet = `dbpedia:Victor_Hugo`
- Prédicat = `dcterms:creator`
- Objet = `dbpedia:Les_Misérables`

Cela donne en RDF/XML :

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:dc="http://purl.org/dc/elements/1.1/"
    xmlns:dcterms="http://purl.org/dc/terms/”
    xmlns:dbpedia="http://dbpedia.org/resource/">
  <rdf:Description rdf:about="dbpedia:Victor_Hugo">
    <dc:title xml:lang="fr">Les misérables</rdfs:label>
    <dcterms:creator rdf:resource="dbpedia:Victor_Hugo"/>
  </rdf:Description>
</rdf:RDF>
```

On peut remarquer que les informations ontologiques (un roman est un type de document, l'année de parution est une date, l'auteur est une personne…) implicitement associées à ces métadonnées n'apparaissent pas au niveau RDF. En effet, la structure de RDF est générique et sert de base à un certain nombre de schémas ou vocabulaires dédiés à des applications spécifiques.

Une partie de ces vocabulaires est spécifiée par le W3C, comme les langages d'ontologie RDFS et OWL, ou le langage SKOS pour la représentation des thésaurus et autres vocabulaires structurés. D'autres vocabulaires RDF, sans être spécifiés par le W3C, sont néanmoins largement utilisés et constituent des standards de fait dans la communauté du Web Sémantique. On peut par exemple citer FOAF (*Friend of a friend*, ou *ami d'un ami*), un vocabulaire RDF constitué de 13 classes avec 62 propriétés permettant de décrire des personnes et le graphe social des relations qu'elles entretiennent entre elles.

L'une des difficultés de la modélisation avec les standards du Web sémantique consiste donc à connaître les ontologies ou vocabulaires déjà définis et à choisir parmi ceux qui pourraient être utilisés dans un contexte donné, pour éviter de tout reconcevoir à partir de zéro. Dans l'exemple

[206] Equivalent à « il existe un objet et il existe un sujet tels que : prédicat(objet, sujet) ».

présenté plus haut, le préfixe `dbpedia` représente une référence à l'ontologie DBpedia (où les concepts `Victor_Hugo` et `Les_Misérables` sont prédéfinis). De même, le préfixe `dcterms` fait référence aux termes définis dans le vocabulaire *Dublin Core*, un schéma de métadonnées qui permet de décrire des ressources et d'établir des relations[207] avec d'autres ressources.

2. RDFS

RDFS (*RDF Schema*) est un langage extensible de représentation des connaissances. RDFS fournit des éléments de base pour la définition d'ontologies ou vocabulaires destinés à structurer des ressources RDF. Les composants principaux de RDFS sont repris dans le langage d'ontologie OWL, qui offre un pouvoir d'expression plus élevé.

a) Classes

`rdfs:Class` permet de déclarer une ressource RDF comme une classe pour d'autres ressources.

Un exemple de `rdfs:Class` est `foaf:Person` dans le vocabulaire FOAF. Une instance de la classe `foaf:Person` est une ressource liée à la classe en utilisant un prédicat `rdf:type`. L'expression formelle suivante traduit en RDFS la phrase en langage naturel : « François est une personne » : `{Francois rdf:type foaf:Person}.`

`rdfs:subClassOf` permet de définir des hiérarchies de classes. Par exemple, « toute personne est un agent » : `{foaf:Person rdfs:subClassOf foaf:Agent}`

b) Propriétés

RDFS précise la notion de propriété définie par RDF en permettant de typer le sujet et l'objet des triplets. Pour cela, RDFS ajoute deux notions :

- `rdfs:domain` définit la classe des sujets liée une propriété, correspondant au domaine de définition d'une fonction.
- `rdfs:range` définit la classe ou le type de données des valeurs de la propriété, donc l'ensemble d'arrivée de la fonction.

RDFS définit aussi les notions de classe, ressource, littéral, propriété, sous-classe, sous-propriété, champ de valeurs et domaine d'application. Par exemple, on pourra exprimer que la propriété « employeur » relie un sujet qui est une personne à un objet qui est une organisation.

```
{employeur rdfs:domain foaf:Person}
{employeur rdfs:range foaf:Organization}
```

À partir de ces déclarations, un système peut déduire de : `{Francois employeur Proxem}` que `Francois` est une instance de `foaf:Person`, et `Proxem` une instance de `foaf:Organization`. On voit ici un point important (mais déstabilisant pour quelqu'un qui aurait déjà pris des habitudes de modélisation avec UML par exemple), à savoir que les mécanismes du Web sémantique sont orientés vers l'inférence et non la vérification des contraintes. De notre point de vue, c'est l'un des facteurs qui complexifie l'adoption de ces standards émergents.

[207] Telles que : titre, créateur, éditeur, sujet, description, langue…

3. SKOS

SKOS (*Simple Knowledge Organisation System*, Système simple d'organisation des connaissances) est une famille de langages formels, construits sur la base de RDF et de RDFS, permettant une représentation standard des taxonomies, thésaurus et autres types de vocabulaires contrôlés. Son développement a été fait conjointement par des acteurs de la communauté RDF et des documentalistes experts.

L'objectif principal de SKOS est de permettre la publication facile de vocabulaires structurés pour leur utilisation dans le cadre du Web sémantique. Nous présentons un exemple de représentation de WordNet avec SKOS page 199.

4. SPARQL

De nombreux langages de requête destinés à interroger les graphes RDF ont été développés. Le langage SPARQL, défini par le W3C, est devenu un standard dans ce domaine. SPARQL définit la syntaxe et la sémantique nécessaires à l'expression de requêtes sur une base de données de type RDF et la forme possible des résultats.

SPARQL est adapté à la structure spécifique des graphes RDF et s'appuie sur les triplets qui les constituent. En cela, il est différent du classique SQL[208], il s'en inspire toutefois clairement dans sa syntaxe et ses fonctionnalités.

SPARQL permet de modifier les données (requêtes `CONSTRUCT`). Les requêtes d'extraction de données (`SELECT`) permettent d'extraire du graphe RDF un sous-graphe correspondant à un ensemble de ressources vérifiant les conditions définies dans une clause `WHERE`. Il a donc aussi des ressemblances avec le langage PROLOG.

5. RIF – SWRL

RIF (*Rule Interchange Format*) est un format d'échange pour les moteurs d'inférences du Web sémantique, permettant de convertir des règles écrites dans des formalismes différents. SWRL (*Semantic Web Rule Language*) est une proposition de langage de règles combinant des éléments d'OWL et de RuleML (Datalog). SWRL permet d'exprimer, en XML, une règle telle que « si on a un parent qui a un frère, alors on a un oncle » :

```
hasParent(?x1,?x2) && hasBrother(?x2,?x3) → hasUncle(?x1, ?x3)
```

C. OWL et les logiques de description

Basé sur une syntaxe RDF, le langage OWL (*Web Ontology Language*) fournit les moyens pour définir des ontologies structurées, c'est-à-dire des terminologies (concepts et propriétés) décrivant des domaines concrets (instance de concepts). OWL étend RDFS pour permettre l'expression de relations complexes entre différentes classes RDFS, ainsi que l'expression de contraintes plus précises sur des classes et des propriétés spécifiques. Cela permet par exemple de :

- Limiter les propriétés d'une classe en termes de cardinalité et de type.

[208] *Structured Query Language*, langage de requête utilisé dans les bases de données relationnelles.

- Induire que les valeurs d'une propriété sont des membres d'une classe particulière ou non.
- Déterminer si tous les membres d'une classe auront une propriété particulière, ou seulement certains d'entre eux.
- Séparer des relations de types un-à-un de relations de type plusieurs-à-un ou un-à-plusieurs, pour représenter des « clés étrangères » d'une base de données dans une ontologie.
- Exprimer des relations entre des classes définies dans différents documents sur le Web.
- Construire de nouvelles classes en dehors de toute union, intersection et complément avec d'autres classes.
- Contraindre un domaine à des combinaisons classe/propriété spécifiques.

L'axiomatique d'OWL se base sur les recherches effectuées dans le domaine des logiques de description (*Description Logics*). OWL définit plusieurs sous-langages, allant du moins expressif (mais garantissant un calcul de preuve rapide) au plus expressif (mais nécessitant éventuellement un temps de calcul dissuasif), selon la logique de description sous-jacente. Notons que le pouvoir d'expression des variantes d'OWL est inférieur à celui d'autres formalismes (comme KIF, *Knowledge Interchange Format*, le langage de la norme *Common Logic*).

Nous allons maintenant présenter les bases mathématiques sous-jacentes au langage OWL, ainsi que ses différentes variantes et les mécanismes de raisonnement qui s'y appliquent. Nous comparerons aussi OWL avec UML (le langage unifié de modélisation).

1. Les logiques de description

a) Concepts de base

Les logiques de description sont une famille de langages de représentation de connaissances utilisés pour formaliser et structurer la connaissance terminologique d'un domaine d'application. Le nom « logique de description » provient des caractéristiques suivantes : d'une part, ces langages définissent leur sémantique formelle en **logique** du premier ordre ; d'autre part, ces langages ont été élaborés pour écrire la **description** des concepts pertinents d'un domaine d'application.

Les logiques de description ont une double ascendance. Elles s'inspirent des réseaux sémantiques de (Quillian, 1968) : des graphes orientés étiquetés dont les nœuds sont des concepts et les arcs des relations. Elles sont aussi influencées par la sémantique des cadres de (Minsky, 1974) : les concepts y sont représentés par des cadres caractérisés par un certain nombre d'attributs (ou *slots*) qui contiennent de l'information sur leur contenu.

Les logiques de description utilisent trois notions de base :

- Les *concepts* correspondent à des « classes d'éléments » (des ensembles dans un univers donné) : `Personne`, `Société`…
- Les *rôles* correspondent aux « liens entre les éléments » (des relations binaires sur un univers donné) : `personneDirigeSociété`, `personneTravailleDansSociété`…
- Les *individus* correspondent aux éléments d'un univers donné : la personne `François`, la société `Proxem`, `leChatDeMaVoisine`…

Ces notions permettent de partitionner la connaissance en deux parties nommées classiquement :

- *T-Box* (axiomes terminologiques), regroupant les concepts et les rôles ; la T-Box définit les règles qui régissent le monde. Ces informations sont « génériques », « globales », vraies pour tous les individus.

- *A-Box* (les individus du monde) : les assertions sont « spécifiques » ou « locales », et s'appliquent à certains individus particuliers.

b) *Survol des familles de logiques de description*

Il existe différentes logiques de description, avec plus ou moins de pouvoir d'expression. Leur base commune est la logique appelée $\mathcal{AL}$. Elle définit les constructeurs suivants : nom de concept, concept *top*, conjonction (« et » logique), quantificateur universel (« quel que soit »), nom de rôle, négation des concepts atomiques.

La logique $\mathcal{AL}$ enrichie de $\mathcal{C}$ (négation de concepts non nécessairement primitifs) donne $\mathcal{ALC}$, qui augmentée par $\mathcal{R}+$ (transitivité des rôles) est notée $\mathcal{S}$. Les langages OWL sont des extensions de cette logique de description $\mathcal{S}$, qui peut ensuite être enrichie par : $\mathcal{H}$ (hiérarchie des rôles), $\mathcal{R}$ (conjonction de rôles), $\mathcal{I}$ (rôles inverses), $\mathcal{O}$ (un-de), $\mathcal{N}$ (restriction de nombre), $\mathcal{Q}$ (restriction de nombre qualifiée), $\mathcal{U}$ (disjonction, le « ou » logique), $\mathcal{E}$ (quantificateur existentiel typé « il existe »), $\mathcal{B}$ (*role filler*).

2. Différents niveaux d'OWL

OWL est la famille des langages de description d'ontologies du Web sémantique. D'autres langages de ce type existent[209] ; on peut considérer qu'OWL a essayé de prendre le meilleur de chacun d'entre eux. Par rapport aux autres langages de description d'ontologies, OWL a la spécificité d'être défini pour être compatible avec l'architecture du Web, en utilisant les URIs pour nommer les objets et RDF pour créer des liens. Les ontologies Web possèdent les avantages suivants :

- Capacité d'être distribuées au travers de nombreux systèmes.
- Capacité de passage à échelle pour les besoins du Web.
- Compatibilité avec les standards pour l'accessibilité et l'internationalisation.
- Ouverture et extensibilité.

OWL Lite a été prévu à l'origine pour des utilisateurs ayant principalement besoin de manipuler une hiérarchie de classification et des contraintes simples. Par exemple, OWL Lite autorise des contraintes de cardinalité, mais seulement avec les valeurs 0 ou 1. OWL Lite est basé sur la logique de description $\mathcal{SHIF}$. Des algorithmes décidables[210] existent pour OWL Lite.

OWL DL a été conçu pour fournir une expressivité maximale tout en garantissant la complétude (le calcul de toutes les conclusions est garanti), la décidabilité (tous les calculs finiront en un temps fini) et des algorithmes de raisonnement implémentables. OWL DL inclut toutes les constructions d'OWL, mais elles ne peuvent être employées que sous certaines restrictions (par exemple, des restrictions de nombre ne peuvent être placées sur les propriétés qui sont déclarées comme transitives). **OWL 2** est une extension d'OWL DL avec des constructeurs supplémentaires qui en simplifient l'usage courant. Les problèmes d'inférence peuvent être en temps exponentiel pour OWL DL ; en pratique,

---

[209] On peut citer par exemple : CycL, Common Logic (standard poussé par l'ISO, http://common-logic.org), DAML+OIL (prédécesseur direct d'OWL), FLogic (http://flora.sourceforge.net), OCML (*Operational Conceptual Modeling Language*, http://technologies.kmi.open.ac.uk/ocml),), XOL (*Ontology Exchange Language*), PowerLoom (http://www.isi.edu/isd/LOOM/PowerLoom), SHOE (*Simple HTML Ontology Extension*, Ontolingua (http://ksl.stanford.edu/software/ontolingua) … La plupart des systèmes qui utilisent DAML, OIL et DAML+OIL sont en train de migrer vers OWL.

[210] Un énoncé est décidable dans une axiomatique si on peut le démontrer ou démontrer sa négation.

les inférences sont souvent effectuées en un temps satisfaisant. OWL DL se fonde sur $\mathcal{SHOIN}$; son évolution OWL 2 est basée sur $\mathcal{SROIQ}$[211].

OWL Full est défini avec une sémantique différente d'OWL Lite ou d'OWL DL, permettant la méta-classification. OWL Full dispose d'un pouvoir d'expression plus élevé et a été conçu pour préserver la compatibilité avec RDFS. Par exemple, en OWL Full une classe peut être traitée simultanément comme une collection d'individus et en tant qu'instance, ce qui n'est pas possible en OWL DL. OWL Full permet à une ontologie d'étendre un vocabulaire prédéfini (RDF ou OWL). Contrairement aux versions précédentes, il n'existe aucun algorithme d'inférence décidable pour OWL Full : une proposition peut y être indémontrable, le contraire de la proposition étant également indémontrable (ou dit autrement, certaines propositions ne peuvent pas y être prouvées).

Comment choisir entre les différentes versions d'OWL ? On aurait pu penser qu'il serait plus simple de fournir des outils pour OWL Lite que pour les autres variantes plus expressives d'OWL, permettant une migration rapide pour les systèmes basés sur des thesaurus et autres taxonomies. En pratique, cependant, la plupart des constructions disponibles en OWL DL peuvent se ramener à des combinaisons complexes des fonctionnalités d'OWL Lite. Le développement d'outils pour OWL Lite s'avérant presque aussi difficile que le développement d'outils pour OWL DL, OWL Lite n'est que peu utilisé. La tendance qui semble se dégager est la prédominance d'OWL 2.

3. Similitudes et différences entre OWL et UML

OWL présente de nombreuses similitudes avec UML. Il est possible d'automatiser la traduction d'un diagramme de classes UML en ontologie (au moins pour en créer la partie taxonomique et le sous-ensemble axiomatique correspondant aux cardinalités de relations).

Néanmoins, la description d'une ontologie et la modélisation d'un système ne recouvrent pas exactement les mêmes objectifs. Une importante différence d'approche réside dans le fait qu'UML décrit essentiellement des classes d'objets, alors qu'une ontologie peut travailler au niveau des instances, en plus des classes. Ainsi, le typage d'une instance doit être explicité en UML alors que, dans une ontologie, il peut être déduit par raisonnements des propriétés de l'instance (notamment des relations qu'elle entretient avec les autres instances). Une autre différence sensible est qu'en UML on exprime des contraintes, alors que les mécanismes d'OWL sont orientés vers l'inférence.

La principale caractéristique commune à UML et OWL est que tous deux sont basés sur la notion de classes. Une classe est un ensemble d'instances ; l'ensemble des instances d'une classe est son extension. Il existe toutefois une subtile différence :

- En UML, l'état d'une instance consiste en un ensemble de propriétés contenant des valeurs d'un type connu.
- En OWL, l'extension d'une classe est un ensemble d'individus (d'instances) qui sont représentées par leur nom ; un individu est défini indépendamment des classes ; il existe une classe ancêtre universel `Thing`, dont l'extension est constituée de tous les individus d'un modèle donné, et toutes les classes héritent de `Thing`.

[211] Une description détaillée est disponible dans *The Even More Irresistible* $\mathcal{SROIQ}$ de Ian Horrocks, Oliver Kutz, et Ulrike Sattler (http://www.cs.man.ac.uk/~sattler/publications/sroiq-TR.pdf).

La principale différence entre OWL et UML, du point de vue des instances, est qu'un individu peut être une instance de Thing mais d'aucune autre classe. Une classe OWL est déclarée en donnant un nom au type considéré : <owl:Class rdf:ID="Societe"/>

Un individu est à la base une ressource RDFS, c'est-à-dire essentiellement un nom. Un individu ID3101 est déclaré de la façon suivante : <owl:Thing rdf:ID="ID3101"/>

En OWL, les relations entre classes sont appelées des propriétés (c'est l'équivalent des rôles d'association dans UML). Les propriétés ne sont pas forcément liées à des classes ; de ce fait, leur nom doit être unique au sein d'un modèle. Par défaut, une propriété est une relation binaire entre Thing et Thing :<owl:ObjectProperty rdf:ID = "travaillePour"/>

Une propriété élémentaire ayant une valeur scalaire est une DatatypeProperty :

```
<owl:DatatypeProperty rdf:ID="NomSociete">
  <rdfs:domain rdf:resource="Societe"/>
  <rdfs:range
    rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
</owl:DatatypeProperty>
```

Une association UML binaire se traduit directement en owl:ObjectProperty. Notons que, comme les associations en UML sont toujours entre types, la propriété OWL a toujours un Domain et un Range.

Une association UML peut être n-aire (plus de deux rôles) ou être une classe associative. Or, en OWL DL, les classes et propriétés sont disjointes (contrairement à ce qu'offre OWL Full). On devra donc convertir chaque relation n-aires UML en un ensemble de n relations binaires en OWL DL avec une classique réification de la relation UML d'origine sous forme de classe OWL.

Une association UML peut être navigable seulement dans un sens ou dans les deux sens. Elle sera convertie en (respectivement) une ou deux propriétés (qui, dans ce dernier cas, seront inverseOf l'une de l'autre).

Les deux langages permettent l'héritage multiple, et de déclarer que les sous-classes d'une classe constituent une partition (exclusion mutuelle). En pratique, OWL rend souvent nécessaire la déclaration de classes disjointes en-dessous d'un ancêtre commun, pour éviter des messages d'erreur lors de l'application des mécanismes d'inférence du raisonneur.

En UML, une association peut avoir une cardinalité minimale et maximale sur chaque rôle. OWL généralise ce mécanisme et permet notamment de déclarer une propriété comme fonctionnelle, inverse-fonctionnelle, symétrique, transitive.

UML a une séparation stricte des niveaux méta[212], alors qu'OWL Full autorise une classe à être une instance d'une autre classe (métaclassification). Enfin, la notion de paquetage UML correspond à la notion d'ontologie en OWL.

---

[212] UML est organisé en une série de méta-niveaux (M3, M2, M1 et M0), de la façon suivante :

- M3 est le MOF (*MetaObject Facility*) défini par l'OMG, langage de modélisation universel.
- M2 est le modèle d'un système de modélisation donné (ex : méta-modèle UML).

4. Principes des raisonneurs

La logique de description permet de réaliser des inférences et des raisonnements. Les tâches de déduction de base sont la subsomption, la vérification d'instances, la vérification de relations, la cohérence de concepts, la cohérence de la base de connaissances. Ces tâches de déduction de base peuvent être utilisées pour définir des tâches plus complexes :

- La recherche permet, étant donné un concept, de trouver les instances de ce concept dans la base de connaissance.
- La réalisation vise, étant donné un individu mentionné dans la base de connaissance, à trouver le concept le plus spécifique dont l'individu est une instance, en accord avec les relations de subsomption.
- La saturation de la *A-Box* sert à compléter les informations sur les individus en accord avec les connaissances de la *T-Box* (concepts et rôles), c'est-à-dire à inférer des propriétés qui sont vraies, même si elles n'ont pas été définies explicitement.

D. Web des données (Linked Data)

Le projet Linked Data vise, comme son nom l'indique, à publier des données structurées non pas en silos indépendants les uns des autres, mais au contraire en les reliant entre elles pour constituer un énorme graphe d'informations.

L'augmentation du nombre de sous-projets parties prenantes dans le Web des données est nette entre juillet 2009 (figure 71) et septembre 2011 (figure 72). Ces sous-projets concernent aussi bien des médias (BBC, New York Times…), des données géographiques (GeoNames, US Census…), des publications (CiteSeer, ACM, projet Gutenberg…), des contenus générés par les utilisateurs (Flickr, Revyu…), des données gouvernementales (NASA, Eurostat, US SEC…), des sources de connaissances à large échelle (DBpedia, FreeBase, OpenCalais, WordNet, YAGO, OpenCyc…) ou les sciences de la vie (PubMed, GeneID…). En septembre 2011, l'ensemble représente 31 milliards de triplets RDF reliés par 504 millions de relations.

C'est là qu'apparaît clairement le grand intérêt du Web sémantique. A partir du moment où une application établit une référence vers l'URI d'une ressource de l'un des sous-projets, l'application peut aussi récupérer de nombreuses autres informations sur les autres sous-projets qui lui sont reliés. Par exemple, le fait de reconnaître une entité nommée de type lieu géographique, puis de lui associer une URI dans DBpedia, permet ensuite d'avoir automatiquement ses coordonnées géographiques dans GeoNames.

- M1 est le modèle d'une application particulière (ex : modèle du diagramme de classe UML).
- M0 est la mise en œuvre d'un modèle M1 dans un cadre concret.

BBC Playcount Data
Surge Radio
LIBRIS
ECS South-ampton
Sem-Web-Central
Wiki-company
RDF ohloh
Resex
MySpace Wrapper
Music-brainz
Audio-Scrobbler
QDOS
Doap-space
Flickr exporter
Semantic Web.org
SW Conference Corpus
Buda-pest BME
Eurécom
ReSIST Project Wiki
IRIT Toulouse
National Science Foundation
BBC Music
Jamendo
BBC Later + TOTP
BBC John Peel
Crunch Base
FOAF profiles
SIOC Sites
Revyu
Open-Guides
ACM
RAE 2001
Pub Guide
Geo-names
Euro-stat
Project Guten-berg
flickr wrappr
Virtuoso Sponger
Pisa
DBLP RKB Explorer
eprints
CORDIS
BBC Programmes
riese
Magna-tune
World Fact-book
Linked MDB
Open Calais
RKB ECS South-ampton
IEEE
New-castle
Linked GeoData
US Census Data
W3C WordNet
DBpedia
lingvoj
Freebase
RDF Book Mashup
DBLP Hannover
CiteSeer
LAAS-CNRS
IBM
UniRef
Gov-Track
UMBEL
GEO Species
DBLP Berlin
Reactome
UniParc
Taxonomy
Open Cyc
Yago
LinkedCT
Drug Bank
PROSITE
Daily Med
Pub Chem
GeneID
Homolo Gene
KEGG
UniProt
Pfam
ProDom
Disea-some
CAS
Gene Ontology
Symbol
OMIM
ChEBI
Inter Pro
UniSTS
HGNC
MGI
PubMed
PDB
As of July 2009

Figure 71 : Sous-projets du Linked Data en juillet 2009

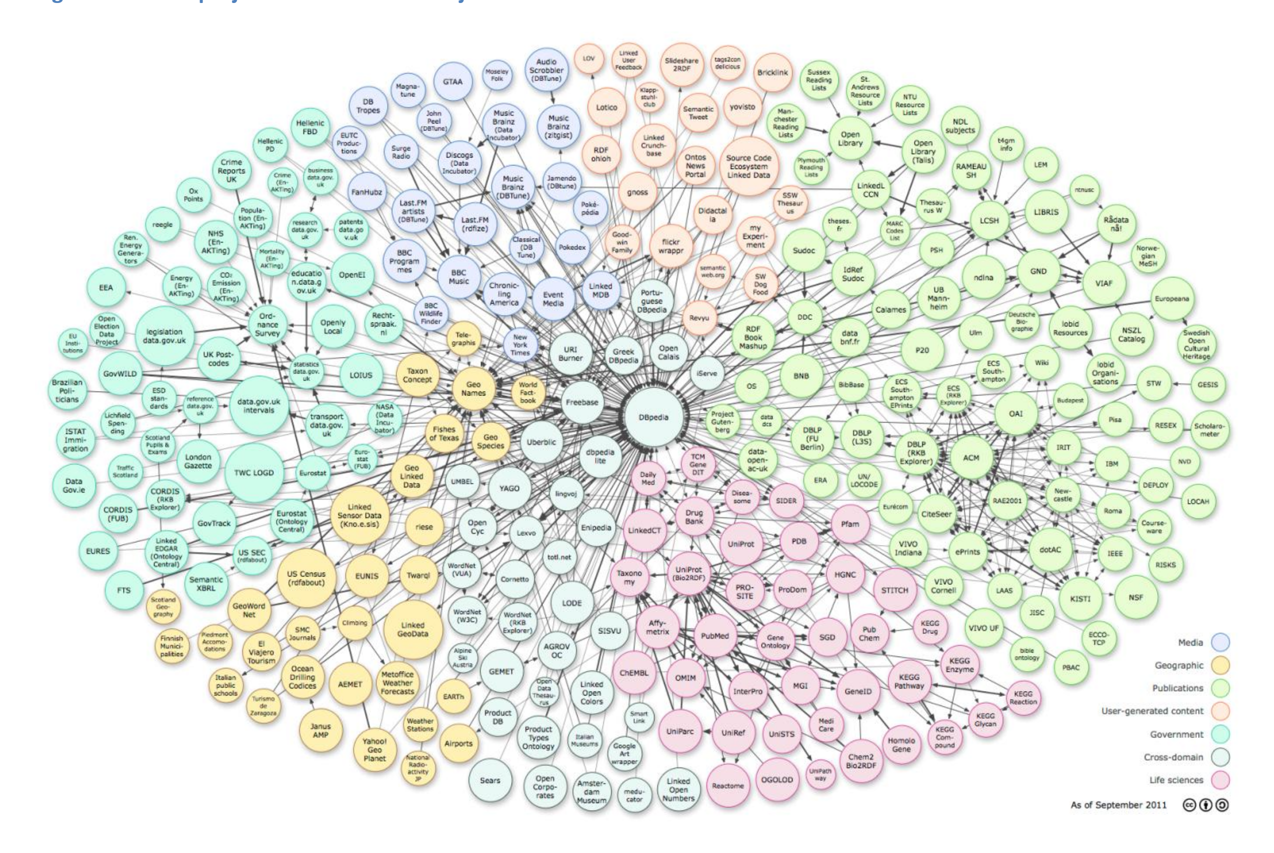


Figure 72 : Sous-projets du Linked Data en septembre 2011

E. Représentation de WordNet avec SKOS

WordNet encode 14 types de relations conceptuelles entre synsets. Les relations structurelles fondamentales sont l'hyponymie / hyperonymie (pour les noms et verbes), traditionnellement représentée par un prédicat `isA`, et la méronymie / holonymie (pour les noms), traditionnellement représentée par un prédicat `hasPart`. SKOS, avec quelques extensions (la définition des types de relations) permet de représenter l'information contenue dans WordNet, ainsi que sa correspondance avec d'autres sources d'informations (comme Wikipédia).

Les lignes suivantes montrent un sous-ensemble d'une telle description. On y voit (après la déclaration des préfixes d'espaces de noms) que :

- La Simple Wikipedia et WordNet 3.0 sont des systèmes de conception.
- WordNet a ENTITY#1 comme concept racine.
- Le concept KITTEN#1 s'appelle usuellement '*kitten*' en anglais et '*chaton*' en français.
- L'hyperonyme du concept KITTEN#1 est le concept YOUNG_MAMMAL#1.
- Les concepts DOMESTIC_CAT#1 de WordNet et CAT de la Wikipedia sont identiques…

```
@prefix skos: <http://www.w3.org/2004/02/skos/core#>
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>
@prefix owl: <http://www.w3.org/2002/07/owl#>
@prefix dct: <http://purl.org/dc/terms/>
@prefix wn: <http://wordnet.princeton.edu/>
@prefix wiki: <http://simple.wikipedia.org/>

wiki:simpleWiki rdf:type skos:ConceptScheme;
  dct:title "Simple English Wikipedia".

wn:wordNet30 rdf:type skos:ConceptScheme;
  dct:title "WordNet 3.0";
  skos:hasTopConcept wn:entity#1.

wn:hasPart rdfs:subPropertyOf skos:related.

wn:kitten#1 rdf:type skos:Concept;
  skos:inScheme wn:wordNet30;
  skos:prefLabel "kitten"@en;
  skos:prefLabel "chaton"@fr;
  skos:broaderTransitive wn:young_mammal#1;

  skos:definition "young domestic cat"@en;
  skos:related wn:domestic_cat#1;
  skos:example "our cat kittened again this year"@en.

wiki:Cat rdf:type skos:Concept;
  skos:inScheme wiki:simpleWiki;
  skos:prefLabel "Cat"@en;
```

```
  skos:definition "Cats, also called domestic cats or house cats,
are carnivorous (meat-eating) mammals, of the family Felidae.".

wiki:Cat skos:closeMatch wn:domestic_cat#1.

wn:violin#1 rdf:type skos:Concept;
  skos:broaderTransitive wn:stringed_instrument#1;
  skos:prefLabel "violin"@en;
  skos:altLabel "fiddle"@en;
  wn:hasPart wn:string#3;
  wn:hasPart wn:fingerboard#3.
```

F. Conclusion

Le Web sémantique offre des promesses fortes. Une migration progressive peut être envisagée entre les formalismes actuellement utilisés et ceux du Web sémantique ; par exemple, les modèles existants (UML) peuvent être dans une grande mesure transformés automatiquement en ontologies. Les données en production au format SQL peuvent également être traduites en RDF. Les pages Web existantes peuvent être progressivement enrichies de métadonnées aux formats du Web sémantique (en utilisant RDFa par exemple).

Toutefois, l'un des enjeux du Web sémantique est de démontrer sa capacité à s'appliquer à une large échelle, et pas seulement dans un cadre restreint. Si les technologies du bas de la « pile » sont éprouvées (Unicode, XML, URI…) ou en train de prendre de l'importance (RDF, RDFS, SPARQL…), celles du haut de la pile (OWL, raisonneurs, SWRL…) ont des spécifications encore fluctuantes et trop peu d'implémentations. Par ailleurs, si les bases de données SPARQL permettent maintenant de faire des requêtes sur des milliards de triplets RDF, la capacité des raisonneurs d'opérer effectivement sur des grands volumes de données ne nous semble pas encore prouvée.

Annexe II. Notions mathématiques

A. Rappel, précision, F-mesure et exactitude

Pour évaluer la qualité d'extraction des entités nommées, nous utilisons les mesures classiques de rappel, de précision et de F-mesure. Nous en rappelons ici la définition.

Le **rappel** correspond à la proportion d'entités nommées correctement trouvées et annotées par rapport au total d'entités nommées réellement présentes dans le texte. Il est donc sensible aux faux négatifs (éléments oubliés ou annotés différemment). Le rappel peut également être interprété non pas comme un ratio mais comme une probabilité : celle qu'une entité nommée sélectionnée aléatoirement soit correctement annotée. Il est défini comme :

$$Rappel_i = \frac{Entités\ correctement\ annotées\ "i"}{Entités\ appartenant\ à\ la\ classe\ "i"}$$

$$Rappel = \frac{\sum_{i=1}^{n} Rappel_i}{n}$$

La **précision** est sensible aux faux positifs (éléments annotés par erreur comme appartenant à une classe donnée). La précision peut, elle aussi, être interprétée comme une probabilité : celle qu'une annotation constatée soit juste. Elle est définie comme :

$$Précision_i = \frac{Entités\ correctement\ annotées\ "i"}{Entités\ annotées\ "i"}$$

$$Précision = \frac{\sum_{i=1}^{n} Précision_i}{n}$$

Aussi appelée *F-score*, la **F-mesure** est la moyenne (harmonique) de la précision et du rappel. En général on utilise la F1-mesure :

$$F = 2 * \frac{(Précision * Rappel)}{(Précision + Rappel)}$$

Pour pondérer l'importance accordée à la précision et au rappel, on utilise la F-Beta mesure :

$$F_\beta = (1 + \beta^2) * \frac{(Précision * Rappel)}{(\beta^2 * Précision + Rappel)}$$

On retrouve relativement fréquemment l'utilisation de cette mesure avec des valeurs de Beta = 0.5 ou Beta = 2. On privilégiera par exemple le rappel dans des tâches où l'on est capable d'effectuer un tri parmi les réponses proposées mais où la perte d'une réponse potentielle est dommageable.

Enfin, l'**exactitude** (*accuracy* en anglais) est le pourcentage des éléments bien classés (des vrais positifs et des vrais négatifs) par rapport à l'ensemble de la population.

B. Algorithme de regroupement spectral

L'algorithme de regroupement spectral prend en entrée une matrice termes-documents A avec *n* documents et *m* termes ; soit M le nombre d'éléments de A différents de zéro. Le point intéressant est que l'algorithme exploite le fait que la matrice est creuse pour optimiser les calculs.

Algorithme de l'étape de division
Entrée: Une matrice A de dimension *n* x *m* **Sortie**: Un arbre avec les lignes de A comme feuilles (1) Soit $\rho \in R^n$ la somme des lignes de AA^T et $\pi = \frac{1}{\sum_i \rho_i} \rho$. (2) Soit R et D des matrices diagonales tels que $R_{i,i} = \rho_i$ et $D_{i,i} = \sqrt{\pi_i}$. (3) Calculer le second plus grand vecteur propre v' de $Q = DR^{-1}AA^T D^{-1}$. (4) Soit $v = D^{-1}v'$ et trier v de façon à ce que $v_i \leq v_{i+1}$. (5) Trouver la valeur t telle que la coupe $(S,T) = (\{1,..,t\},\{t+1,..,n\})$ minimise la conductance : $$\phi(S,T) = \frac{c(S,T)}{\min(c(S),c(T))}$$ où $c(S,T) = \sum_{i \in S, j \in T} A_{(i)} \cdot A_{(j)}$ et $c(S) = c(S,\{1,..,n\})$ (6) Soit A_S et A_T les sous matrices de A. Répéter les étapes 1 à 5 sur les matrices A_S et A_T.

Les étapes (2) à (5) proviennent d'un résultat de la théorie spectrale qui indique que la recherche d'un bipartitionnement revient à la recherche du second plus grand vecteur propre (le vecteur propre associé à la seconde plus grande valeur propre de la matrice Laplacienne du graphe). Les autres étapes permettant d'éviter de calculer cette matrice elle-même.

Ensuite l'algorithme de fusion reprend l'arbre généré et regroupe les feuilles de ce dernier pour avoir un regroupement optimal dans l'arbre.

Algorithme de l'étape de fusion
Entrée: Un arbre avec les lignes de A comme feuilles **Sortie:** Un ensemble de regroupement des lignes de A (1) Pour chaque feuille de l'arbre créé un regroupement C_i (2) Pour chaque nœud on calcule : g(C_n) où $C_n = C_g \cup C_d$ et g(C_g)+g(C_d) avec C_g et C_d les regroupements optimaux dans l'arbre pour les fils gauche et droite du nœud *n*. et g étant une fonction évaluant la qualité des regroupements : $\sum_i \alpha(\sum_{u,v \in C_i} 1 - A_{(u)} \cdot A_{(v)}) + \beta(\sum_{u \in C_i, v \notin C_i} A_{(u)} \cdot A_{(v)})$ où $A_{(i)}$ est la i^{ème} ligne de A (3) Si g(C_i)>g(C_g)+g(C_d) alors les regroupements C_g et C_d sont fusionnés sinon on s'arrête

Pour l'implémentation de l'algorithme de fusion, nous utilisons une structure permettant de garder les valeurs de la fonction ***g*** ainsi que de ses composantes pour chaque nœud. Les formules suivantes sont utilisées pour calculer g(C_n) de l'étape (2) :

$$\sum_{u\in C_n, v\notin C_n} A_{(u)} \cdot A_{(v)} = \sum_{u\in C_g, v\notin C_g} A_{(u)} \cdot A_{(v)} + \sum_{u\in C_d, v\notin C_d} A_{(u)} \cdot A_{(v)} - 2 \cdot \sum_{u\in C_g, v\in C_d} A_{(u)} \cdot A_{(v)}$$

Ce qui revient à la formule suivante si on note $\beta_n = \sum_{u\in C_n, v\notin C_n} A_{(u)} \cdot A_{(v)}$

$$\beta_n = \beta_g + \beta_d - 2 \cdot diff$$

De la même manière :

$$\sum_{u,v\in C_n} 1 - A_{(u)} \cdot A_{(v)} = \sum_{u,v\in C_g} 1 - A_{(u)} \cdot A_{(v)} + \sum_{u,v\in C_d} 1 - A_{(u)} \cdot A_{(v)} + \sum_{u\in C_g, v\in C_d} 1 - A_{(u)} \cdot A_{(v)}$$

Ce qui revient à la formule suivante si on note $\alpha_n = \sum_{u,v\in C_n} 1 - A_{(u)} \cdot A_{(v)}$

$$\alpha_n = \alpha_g + \alpha_d + |diff| - diff$$

Ainsi le calcul de ***g*** est plus efficace et revient au calcul de ***diff*** *qui correspond* à un seul élément pour deux feuilles.

C. Les CRF

1. Présentation

Les CRF sont des modèles graphiques non dirigés ayant pour objectif de définir une distribution de probabilités sur les annotations Y (les classes d'entités nommées) étant donnée une observation X (la séquence de mots). Ils sont définis comme suit. Soit $G = (V, E)$ un graphe non dirigé (appelé graphe d'indépendances) où V est l'ensemble des nœuds et E l'ensemble des arcs, et X et Y deux champs aléatoires décrivant respectivement l'observation et son annotation, de sorte que pour chaque nœud v (pris dans V) il existe une variable aléatoire Y_v dans Y. On dit que (X, Y) est un champ conditionnel aléatoire si chaque variable aléatoire Y_v respecte la propriété de Markov suivante :

$$\forall v, p\ (Y_v | X, \{Y_w, w \neq v\}) = p(Y_v | X, Y_w, (V, W) \in E\})$$

C'est-à-dire que chaque variable aléatoire Y_v dépend uniquement de X et de ses voisins dans le graphe d'indépendances. D'après le théorème de Hammersley-Clifford (Hammersley, Clifford, 1971), cette condition d'indépendance permet d'écrire la probabilité d'une annotation y, étant donnée une observation x, comme un produit de fonctions de potentiel $\psi(y\ , x)$ sur tous les sous-graphes complètement connectés (i.e. les cliques) du graphe d'indépendances.

$$p(y|x) = \frac{1}{Z(x)} \prod_{c \in C} \psi_c(\, y_c \,, c)$$

où :

- C est l'ensemble des cliques de G.
- y_c est la configuration prise par les variables aléatoires de la clique c dans l'observation y.
- $Z(x)$ est un coefficient de normalisation défini comme suit :

$$Z(x) = \sum_{y} \prod_{c \in C} \psi_c(y_c, x)$$

Pour les CRF, (Lafferty, McCallum et Pereira, 2001) ont proposé de définir la forme de ces fonctions de potentiel comme l'exponentielle d'une somme pondérée de fonctions f_k appelées *features* (ou *fonctions caractéristiques*), les λ_k étant les poids associés a chacune de ces caractéristiques :

$$\psi_c(\, y_c \,, c) = exp\left(\sum_{k} \lambda_k \, f_k(y_c, x, c)\right)$$

Les caractéristiques sont des fonctions à valeurs réelles, mais dans la plupart des cas elles sont simplement des fonctions binaires, valant 1 si un phénomène donné est observé, 0 sinon. C'est à travers elles que toutes les connaissances du domaine sont intégrées dans le modèle. Ces caractéristiques prennent en paramètres les valeurs prises par les variables aléatoires de la clique sur laquelle elles s'appliquent (y_c), ainsi que l'ensemble de l'observation x. Par conséquent, la valeur prise par une variable aléatoire peut dépendre de toute l'observation x. Par exemple, dans le cas de l'annotation d'une séquence, le choix de l'étiquette associée au dernier élément de la séquence peut être lié à la valeur du premier élément de cette séquence.

A ces caractéristiques sont associés des poids λ_k. Ces poids sont les paramètres du modèle. Ils permettent d'attacher plus ou moins d'importance à certaines caractéristiques, ou même d'indiquer que le phénomène caractérisé par une *feature* ne doit pas se produire (si le poids est négatif).

Un CRF est donc défini par :

- Un graphe d'indépendances G.
- Un ensemble de caractéristiques f_k, auxquelles sont associés des poids λ_k.

La probabilité conditionnelle d'une annotation connaissant une observation, telle que définie par un CRF, s'exprime alors par :

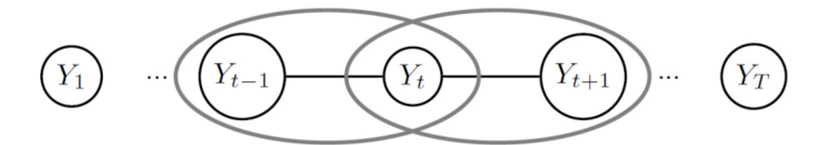
$$p(y|x) = \frac{1}{Z(X = x)} exp\left(\sum_{c \in C} \sum_{k} \lambda_k \, f_k(y_c, x, c)\right)$$

où Z(x) se réécrit :

$$Z(x) = \sum_{y} exp\left(\sum_{c \in C} \sum_{k} \lambda_k \, f_k(y_c, x, c)\right)$$

Le premier problème associé aux CRF est celui de l'annotation, qui consiste à rechercher l'annotation la plus probable associée à une observation. Le second problème est celui de l'inférence ou de l'apprentissage du CRF, qui consiste à estimer les paramètres λ_k qui maximisent la vraisemblance du modèle par rapport à un échantillon d'observations annotées. Ces paramètres peuvent être appris en utilisant une méthode classique de maximisation de la log-vraisemblance. Les paramètres optimaux ne pouvant pas être calculés de façon analytique, des méthodes de descente de gradient sont utilisées. La plus performante dans ce contexte semble être l'algorithme BFGS à mémoire limitée (L-BFGS).

Dans la littérature, les CRF ont pour l'instant été utilisés essentiellement dans le cas de l'annotation de séquences (même si des travaux récents portent sur l'apprentissage sur des structures d'arbres). Dans ces travaux sur les séquences, le graphe d'indépendances utilisé est une chaîne linéaire du premier ordre :



Dans ce type de modèle, la probabilité d'une annotation s'exprime comme suit :

$$p(y|x) = \frac{1}{Z(x)} exp\left(\sum_{t=2}^{T}\sum_{k} \lambda_k f_k(y_{t-1}, y_t, x, t)\right)$$

où T est la longueur de la séquence x. Les caractéristiques sont de la forme$f_k(y_{t-1}, y_t, x, t)$ car les cliques du graphe d'indépendances sont les paires de nœuds (y_{t-1}, y_t). On note ici que, par souci de simplification, on ne considère que les cliques à deux nœuds dans l'écriture des formules et algorithmes. En effet, les cliques à un nœud peuvent être traitées de façon similaire.

L'intérêt principal de travailler avec un graphe d'indépendances aussi simple que celui-ci est de permettre de mettre en œuvre des techniques de programmation dynamique pour un calcul efficace des deux tâches principales des CRF, que sont (a) la recherche de l'annotation la plus probable et (b) l'apprentissage des paramètres du modèle. Nous donnons ici un aperçu de ces deux algorithmes.

2. Recherche de l'annotation la plus probable

La recherche de l'annotation la plus probable consiste à trouver l'annotation y maximisant la probabilité $p(y|x)$, étant donnés une observation x et un CRF dont les paramètres λ_k sont connus.

$$\hat{y} = \arg max\, p(y|x) = \arg max \left(\sum_{t=2}^{T}\sum_{k} \lambda_k f_k(y_{t-1}, y_t, x, t)\right)$$

Il est évidemment impossible de calculer naïvement cette valeur pour toutes les annotations y possibles. Toutefois, la forme du graphe permet de mettre en place l'algorithme de Viterbi[213]. Pour cela, on définit le coefficient $\delta_t(y_t)$ comme étant le « score » (la somme pondérée des caractéristiques sur toute la séquence) de la meilleure annotation de x_1 ... x_t où x_t est annoté par y_t. Sa formule de récurrence est définie comme suit :

$$\delta_{t+1}(y_t + 1) = \max \delta_t(y_t) \exp(\sum_k \lambda_k f_k(y_{t-1}, y_t, x, t+1))$$

Cette variable permet d'obtenir aisément le score de la meilleure annotation :

$$\max \sum_{t=2}^{T} \sum_k \lambda_k f_k(y_{t-1}, y_t, x, t) = \max \delta_T(y_T)$$

En mémorisant le chemin de Viterbi correspondant, on obtient la meilleure annotation y de l'observation x.

3. Apprentissage des paramètres du modèle

La tâche d'apprentissage d'un CRF consiste, étant donné un ensemble d'apprentissage S = {(x(1), y (1)), ..., (x(N), y(N))}, à trouver les poids Λ qui maximisent la log-vraisemblance du modèle :

$$L_\Lambda = \sum_{i=1}^{N} log\left(p_\Lambda\left(y^{(i)}|x^{(i)}\right)\right)$$

où $p_\Lambda(y\,|x\,)$ est la probabilité de l'annotation y sachant l'observation x dans le CRF dont les paramètres sont Λ. L'algorithme de maximisation de la log-vraisemblance utilisé étant une descente de gradient (L-BFGS), L_Λ est calculé à chaque étape avec des paramètres Λ différents.

Il est donc nécessaire de calculer cette fonction efficacement. La partie coûteuse de ce calcul est le coefficient de normalisation Z(x) qui intervient dans la probabilité $p\,(y\,|x\,)$. En effet, celui-ci est la somme des probabilités non normalisées (p(y|x) × Z(x)) pour toutes les annotations possibles de x.

Une méthode de programmation dynamique est donc employée. Pour cela, on définit les coefficients *forward* $\alpha_t(y_t)$. $\alpha_t(y_t)$ est la probabilité non normalisée de toutes les annotations possibles de la séquence x_1 ... x_t où x_t est annoté par y_t . La formule de récurrence de ce coefficient est :

$$\alpha_{t+1}(y_{t+1}) = \sum_{y_t} \alpha_t(y_t) \exp\left(\sum_{\mathrm{k}} \mathrm{\lambda_k f_k(y_t, y_{t+1}, x, t+1)}\right)$$

[213] L'algorithme de Viterbi est un algorithme de programmation dynamique très utilisé dans le traitement des séquences à états cachés.

En utilisant ce coefficient, on peut calculer Z(x) de la façon suivante :

$$Z(x) = \sum_{yT} \alpha_T(y_T)$$

Le lecteur trouvera une introduction pratique à l'utilisation des CRF pour annoter des séquences dans (Truyen, Phung, 2008), qui propose comme étude de cas l'annotation des syntagmes nominaux dans un texte.

Annexe III. Références linguistiques

A. Liste des rôles thématiques de VerbNet

Nous présentons ici l'ensemble des rôles

1. Acteur

Acteur est utilisé dans des classes de communication ("**chitchat**", "**marry**", "**meet**") quand les deux arguments peuvent être considérés comme symétriques, comme dans « Pierre et Marie se fiancent ».

2. Agent

Agent est un instigateur actif d'une action ou d'un événement. *Agent* est généralement un humain ou un sujet animé ; il peut aussi être utilisé pour désigner un sujet ayant une volonté propre, comme une force ou une machine.

Pour identifier un rôle *Agent*, on peut utiliser :

- Le test de volonté (« Tom cassa volontairement la tasse » vs. « *Tom se sentit malade volontairement »).
- Le test de promesse (« Tom promit de casser la tasse » vs. « *Tom promit de se sentir malade »).

3. Attribut

Attribut de *Patient* ou de *Thème* fait référence à une caractéristique de quelque chose qui est en train de changer, comme dans « le prix du pétrole augmente ». *Attribut* a une contrainte de sélection de type scalaire (définie par une quantité, une masse, une longueur, une heure, une température, etc.).

4. Bénéficiaire

Bénéficiaire désigne l'entité bénéficiant d'une action, généralement introduite par une proposition commençant par « pour », comme dans « Marie a créé un jouet pour le bébé » ou « donner quelque chose à quelqu'un ».

5. Cause

Cause est surtout utilisé par les classes de verbes psychologiques ou relatifs au corps, comme dans « les touristes ont admiré les tableaux » ou « cela compte pour moi ».

6. Destination

Destination est le point final ou la direction d'un déplacement (introduit par « vers » ou « sur »…). Il est utilisé dans des classes telles que "**banish**", "**send**", "**carry**", comme dans « le roi a exilé le capitaine sur l'île ».

7. Emplacement

Emplacement est un participant qui exprime une destination, une origine, un endroit, généralement introduit par un complément circonstanciel de lieu, comme dans « le bateau apparaît à l'horizon ».

8. Étendue

Le rôle *Étendue* est utilisé pour spécifier l'intervalle ou le degré de changement, comme dans « le prix du pétrole augmente de 10 % ».

9. Expérimentateur

Expérimentateur est un participant caractérisé par le fait d'avoir conscience de quelque chose ou d'expérimenter quelque chose, comme dans « Pierre souffre ». Plusieurs verbes psychologiques ou d'émotion ont un *Expérimentateur* pour sujet (aimer, admirer…) ou pour objet (amuser, perturber…).

10. Moment

Moment est un rôle spécifique à la classe "**begin**" pour exprimer un horaire, comme dans « la réunion commence à 16 heures », « Pierre arrive dans 3 jours ».

11. Instrument

Instrument est un objet ou une force physique qui provoque un changement dans quelque chose, généralement par contact direct, comme dans « il toucha la balle avec la raquette ».

12. Matériau

Matériau est le point de départ d'une transformation, utilisé par exemple dans "**build**", comme dans « Marie a sculpté une jolie statuette avec le bout de bois ».

13. Montant

Montant est utilisé pour représenter une valeur, une somme d'argent ou l'équivalent (par exemple dans les classes "**build**", "**get**", "**obtain**"), comme dans « ils lui ont facturé 10 € ».

14. Objectif

Objectif est le participant vers lequel le mouvement a lieu, comme par exemple dans « les martiens rentrent à la maison ».

15. Patient

Patient est un participant soumis à un processus ou affecté par une action. L'emphase est mise sur le changement d'état. Le *Patient* peut être sujet (« la glace a fondu ») ou objet du verbe (« il chauffa

l'eau »). *Patient1* et *Patient2* sont aussi utilisés en cas de rôles symétriques (« la crème et l'œuf se mélangèrent »).

Pour déterminer un rôle *Patient*, un test possible est « qu'est-ce qui est arrivé à X ? ».

16. Prédicat

Prédicat est la partie de l'énoncé qui exprime ce qui est dit à propos du *Thème*, comme dans « il se vante d'être l'homme le plus fort du monde ».

17. Produit

Produit est le résultat final d'une transformation, comme dans « David a construit une maison ».

18. Récipient

Récipient est un participant qui est la destination du transfert d'une entité concrète ou abstraite, comme dans « Jean a passé le sel à Marie ». Ce rôle autorise toujours une contrainte de sélection de type Animé et parfois Organisation. On remarquera que la frontière avec le *Bénéficiaire* semble floue.

19. Source

Source est le point de départ du mouvement, généralement introduit par une préposition (« les martiens viennent d'une autre planète »).

20. Stimulus

Stimulus est l'événement ou l'objet qui provoque une réaction chez un *Expérimentateur*, comme dans « l'orage effraya les enfants ». Ce rôle n'impose généralement pas de contrainte de sélection.

21. Thème

Thème est un participant qui est localisé dans un endroit ou qui se déplace d'un endroit à l'autre. L'emphase est mise sur la localisation ou la possession. (« Jean donne un ballon », « Marie marche »…)

Thème1 et *Thème2* peuvent être utilisés en cas de rôles symétriques, comme dans « Jean échange le livre pour une revue ».

22. Topique

Topique est le rôle utilisé par les verbes de communication pour exprimer le thème ou le sujet d'une conversation ou d'un transfert de message, comme dans « Pierre a mis en garde Marie contre les effets de la colère ».

Index